



TIMO BERTHOLD<sup>1</sup>

JAKOB WITZIG<sup>2</sup>

## Conflict Analysis for MINLP

---

<sup>1</sup>  0000-0002-6320-8154  
<sup>2</sup>  0000-0003-2698-0767

Zuse Institute Berlin  
Takustr. 7  
14195 Berlin  
Germany

Telephone: +49 30-84185-0  
Telefax: +49 30-84185-125

E-mail: [bibliothek@zib.de](mailto:bibliothek@zib.de)  
URL: <http://www.zib.de>

ZIB-Report (Print) ISSN 1438-0064  
ZIB-Report (Internet) ISSN 2192-7782

# Conflict Analysis for MINLP

Timo Berthold and Jakob Witzig

FICO Germany GmbH, Takustr. 7, 14195 Berlin, Germany  
timoberthold@fico.com

Zuse Institute Berlin, Takustr. 7, 14195 Berlin, Germany  
witzig@zib.de

July 15, 2020

## Abstract

The generalization of MIP techniques to deal with nonlinear, potentially non-convex, constraints have been a fruitful direction of research for computational MINLP in the last decade. In this paper, we follow that path in order to extend another essential subroutine of modern MIP solvers towards the case of nonlinear optimization: the analysis of infeasible subproblems for learning additional valid constraints. To this end, we derive two different strategies, geared towards two different solution approaches. These are using *local* dual proofs of infeasibility for *LP-based* branch-and-bound and the creation of *nonlinear* dual proofs for *NLP-based* branch-and-bound, respectively. We discuss implementation details of both approaches and present an extensive computational study, showing that both techniques can significantly enhance performance when solving MINLPs to global optimality.

## 1 Introduction

Mixed integer nonlinear programs (MINLPs) are arguably among the hardest optimization problems, with a wide range of applications in Chemical Engineering (e.g., [Biegler et al. 1997](#), [Floudas 2000](#), [Kallrath 2005](#), [Witzig et al. 2018](#)), Computational Biology (e.g., [Phillips and Rosen 1994](#), [Liberti et al. 2008, 2009](#)), Portfolio Optimization (e.g., [Cornuéjols and Tütüncü 2006](#)) and many others.

MINLP solvers that are based on linear relaxations and spatial branching work similarly to mixed integer programming (MIP) solvers in the sense that they are based on a branch-and-cut algorithm, enhanced by various heuristics, domain propagation, and presolving techniques. However, the analysis of infeasible subproblems, which is an important component of most major MIP solvers, has been hardly studied in the context of MINLPs. There are two

main approaches for analyzing infeasibility in MIP solvers: *conflict graph analysis* (Achterberg 2007a), which originates from artificial intelligence and constraint programming, and *dual proof analysis* (e.g., Witzig et al. 2019a). Both together are often subsumed under the term *conflict analysis*.

In this paper, we consider MINLPs of the form

$$\min_{x \in X} \{f(x) \mid g_k(x) \leq 0 \forall k \in \mathcal{K}, h_e(x) = 0 \forall e \in \mathcal{E}, x_i \in \mathbb{Z} \forall i \in \mathcal{I}\}, \quad (1)$$

with convex objective function  $f: X \mapsto \mathbb{R}$ , nonlinear constraint functions  $g_k: X \mapsto \mathbb{R}$ ,  $k \in \mathcal{K} := \{1, \dots, p\}$ , continuously differentiable, possibly nonconvex, and affine functions  $h_e: X \mapsto \mathbb{R}$ ,  $e \in \mathcal{E} := \{1, \dots, q\}$ . Moreover, let  $X \subseteq \mathbb{R}^n$  be a non-empty convex set, let  $\mathcal{N} = \{1, \dots, n\}$  be the index set of all variables and  $\mathcal{I} \subseteq \mathcal{N}$  the set of variables that need to be integral in every feasible solution. We call an MINLP *convex* if all of its constraint functions  $g_k$  are convex. Otherwise, we call the MINLP *nonconvex*. If  $f$  is a linear function and all functions  $g_k, h_e$  are affine, we call (1) a *mixed integer program* (MIP).

For a general MINLP (1), we obtain its *nonlinear programming* (NLP) relaxation by omitting the integrality requirements. The MIP relaxation of an MINLP with a linear objective<sup>1</sup> is given by omitting all “truly” nonlinear constraints  $g_k, h_e$  and keeping only those that are representable as a linear constraint matrix, eventually enhanced by linear underestimators of some of the nonlinear constraints. Omitting both, integrality requirements and truly nonlinear constraints, yields the *linear programming* (LP) relaxation.

All three relaxations provide a lower bound on the optimal solution value of a MINLP of form (1). In theory, linear and convex smooth nonlinear programs are solvable in polynomial time (Khachiyan 1979, Vavasis 1995). Also in practice, both classes can be solved very efficiently (Bixby 2002, Nocedal and Wright 2006). In contrast to that, nonconvexities as imposed by integer variables or nonconvex nonlinear functions easily lead to problems that are both  $\mathcal{NP}$ -hard in theory and computationally demanding in practice.

In this paper, we focus on solving MINLPs either by LP-based or by NLP-based branch-and-bound. During a branch-and-bound search, roughly half of the considered subproblems are either found to be infeasible or suboptimal, i.e., to exceed the bound given by the best solution found so far (which can be seen as a special case of infeasibility). In contrast to modern MIP solvers that can refer to a variety of well-studied techniques to “learn” from infeasible and bound-exceeding subproblems (e.g., Davey et al. 2002, Sandholm and Shields 2006, Achterberg 2007b, Witzig et al. 2017, 2019a), similar techniques for MINLPs exist for certain special cases only.

To close this gap, we introduce two variants of conflict analysis for general MINLP. The first is a straightforward generalization of MIP concepts to LP-based branch-and-bound for MINLP. Our additional contribution is the use of locally valid certificates of infeasibility to deal with locally valid linear approximations of nonconvex constraints. The second technique is a generalization of

<sup>1</sup>It can be assumed w.l.o.g. that the objective of (1) is linear since a nonlinear objective can be transformed into a constraint bounded by an artificial variable that is minimized.

the theory of Farkas proofs to work with nonlinear relaxations in an NLP-based branch-and-bound. We show how we can use the dual multipliers of an infeasible convex NLP to generate a valid linear constraint. This paper is the full version of a short conference proceedings paper (Witzig et al. 2019b). While in Witzig et al. (2019b), we only briefly sketched the idea of nonlinear dual proofs, in the present paper we give a more detailed theoretical background and present a working implementation of the concept in a state-of-the-art MINLP solver. We discuss various implementation details and present a thorough computational study of both techniques.

## 2 Background and Related Work

In this section, we will briefly discuss different algorithms and their implementations for solving MINLPs, provide the reader with the technical background of conflict analysis in MIP, and review related work on infeasibility in MINLP solving.

### 2.1 Solving MINLPs

Commonly used methods to solve convex MINLPs include the extended cutting plane algorithm (ECP) (Westerlund and Pettersson 1995), the extended supporting hyperplane algorithm (Kronqvist et al. 2018), outer approximation (OA) (Duran and Grossmann 1986, Fletcher and Leyffer 1994), NLP-based branch-and-bound (Gupta and Ravindran 1985), and LP/NLP-based branch-and-bound (Quesada and Grossmann 1992). The most commonly used method to solve nonconvex MINLPs is a combination of either OA or ECP (Kocis and Grossmann 1988, Viswanathan and Grossmann 1990) and spatial branch-and-bound (Land and Doig 1960, Liberti and Pantelides 2003, Horst and Tuy 2013). Different MINLP solvers either use LP or MIP relaxations or both during the tree search. For example, Couenne<sup>2</sup> (Belotti et al. 2009) and SCIP<sup>3</sup> (Vigerske and Gleixner 2018) derive valid lower bounds by solving LP relaxations only, whereas ANTIGONE<sup>4</sup> (Misener and Floudas 2014), BARON<sup>5</sup> (Kılınç Karzan et al. 2009) and BONMIN<sup>6</sup> (Bonami et al. 2008) solve both LP and MIP relaxations. In contrast to that, only a handful of MINLP solvers provide the possibility to exclusively use NLP relaxations, e.g., BONMIN and FICO Xpress<sup>7</sup> (Belotti et al. 2016). Notably, BONMIN by default chooses a hybrid strategy between LP-based and NLP-based branch-and-bound (Bonami et al. 2008), demonstrating that ideally, new solving techniques are developed in a way that they can be incorporated into either paradigm. For a detailed overview of MINLP solvers that can handle convex and/or nonconvex MINLPs and the implemented algorithms,

---

<sup>2</sup><https://github.com/coin-or/Couenne>

<sup>3</sup><https://www.scipopt.org>

<sup>4</sup><http://ares.tamu.edu/ANTIGONE/>

<sup>5</sup><https://minlp.com/baron>

<sup>6</sup><https://github.com/coin-or/Bonmin>

<sup>7</sup><https://www.fico.com/en/products/fico-xpress-solver>

we refer to (Kronqvist et al. 2018).

In the following, we will focus on MINLP solvers that use a combination of LP-based branch-and-bound (e.g., generated by an ECP) and spatial branch-and-bound. Spatial branch-and-bound is – as a special variant of LP-based branch-and-bound (Dakin 1965, Land and Doig 1960) – a divide-and-conquer method which splits the search space sequentially into smaller and smaller subproblems that are intended to be easier to solve. Additionally, convex relaxations are used to compute lower bounds on the individual subproblems. A subproblem can be pruned if the lower bound exceeds the currently best-known solution. To divide the search space into smaller pieces, spatial branch-and-bound branches on integer variables with a fractional solution value in the relaxation solution. In addition to that, spatial branch-and-bound can branch on variables (possibly continuous) if they appear in nonconvex terms of nonlinear constraints that are violated by the current relaxation solution.

During a branch-and-bound procedure, infeasible subproblems may be encountered. Infeasibility can either be detected by an infeasible relaxation or by contradicting variable bounds, derived by domain propagation. The goal of conflict analysis is to learn from these infeasibilities, in order to prune the search tree more efficiently going forward.

## 2.2 Conflict Analysis in MIP

To a certain extent, conflict analysis techniques for MIP can also be applied within MINLP solvers that rely on LP-based branch-and-bound. Hence, we will look at MIP conflict analysis techniques in this subsection and discuss their limitations when carried-over one-to-one for solving MINLPs in the next subsection.

Here and in Section 3, we will assume the objective function of (1) to be linear. This happens without loss of generality, since a nonlinear objective function can be transformed into a constraint bounded by an artificial variable  $z$  that is minimized.

Furthermore, whenever we will use standard MIP notation whenever considering a MIP or an LP. Therefore, we will denote the linear objective as  $f(x) = c^\top x$ . Moreover, we treat the set of constraints  $g_k(x) \leq 0, h_e(x) = 0$  as a constraint matrix of form  $Ax \leq b$  by considering  $-h_e(x) \leq 0$  and  $h_e(x) \leq 0$  and moving the (potentially nonzero) constants of the affine functions to the right-hand side of the constraint matrix. Finally, the set  $X$  is represented by the intersection of variable bounds  $\ell \leq x \leq u$  with  $\ell, u \in \overline{\mathbb{R}}^n$ , where  $\overline{\mathbb{R}} := \mathbb{R} \cup \{\pm\infty\}$ . Hence a MIP reads as

$$\min\{c^\top x \mid Ax \leq b, \ell \leq x \leq u, x_j \in \mathbb{Z} \forall j \in \mathcal{I}\}. \quad (2)$$

Conflict analysis for MIPs has a long history, having its origin in artificial intelligence (Stallman and Sussman 1977) and solving satisfiability problems (SAT) (Marques-Silva and Sakallah 1999). Similar ideas are used in constraint

programming (CP) (e.g., Ginsberg 1993, Jiang et al. 1994). Integrations of these techniques into MIP were independently suggested in (Davey et al. 2002, Sandholm and Shields 2006, Achterberg 2007a).

If infeasibility is encountered by domain propagation, modern SAT and MIP solvers construct a directed acyclic graph which represents the logic of how the set of branching decisions led to the detection of infeasibility. This graph is called the *conflict graph*. Valid *conflict constraints* can be derived from cuts in the graph that separate the branching decisions from an artificial vertex representing the infeasibility. Based on such a cut, a conflict constraint consists of a set of variables with associated bounds, requiring that in each feasible solution at least one of the variables has to take a value outside these bounds.

If the LP relaxation of a subproblem with local bounds  $\ell'$  and  $u'$  turns out to be infeasible, it is necessary to identify a set of variables and bound changes that are sufficient to render the infeasibility. Such a set, the so-called *dual proof constraint*, see, e.g., Witzig et al. (2017, 2019a), can be constructed by using LP duality theory that states that exactly one of the systems

$$Ax \leq b, \ell' \leq x \leq u' \quad (3)$$

$$y^T A + r^u - r^\ell = 0, y^T b + \sum_{j \in \mathcal{N}} (r_j^u u'_j - r_j^\ell \ell'_j) < 0, y, r^\ell, r^u \geq 0 \quad (4)$$

can be satisfied. System (4) implies a proof of infeasibility with respect to the local bounds

$$\begin{aligned} 0 &> y^T b + \sum_{j \in \mathcal{N}} (r_j^u u'_j - r_j^\ell \ell'_j) \\ \iff 0 &> y^T b - (y^T A)\{\ell', u'\} \\ \iff (y^T A)\{\ell', u'\} &> y^T b, \end{aligned}$$

where  $A_{.j}$  denotes the  $j$ -th column of constraint matrix  $A$  and  $(y^T A)\{\ell', u'\} := \sum_{j \in \mathcal{N}: y^T A_{.j} > 0} (y^T A_{.j}) \ell'_j + \sum_{j \in \mathcal{N}: y^T A_{.j} < 0} (y^T A_{.j}) u'_j$ . Thus,  $(y^T A)\{\ell', u'\}$  is equivalent to the minimal activity with respect to the local bounds  $\ell'$  and  $u'$ . Consequently, every feasible solution has to satisfy

$$(y^T A)x \leq y^T b, \quad (5)$$

which is called a *dual proof constraint*; it is a globally valid constraint because it is a conic combination of all globally valid constraints. Thereby, dual proof constraints are a special case of Benders cuts (Benders 1962). The (initial) dual proof constraint is used as a starting point for conflict graph analysis or dual proof analysis. As defined by Witzig et al. (2019a), *dual proof analysis* denotes the non-trivial modification of dual proof constraints, e.g., by lifting or elimination techniques. Both, the constraints resulting from dual proof analysis and conflict constraints, are typically used for propagation in the remainder of the MIP search, but they are not added to the LP relaxation.

Note that in MIP, conflict graph analysis might yield several conflicts per infeasibility. These might be of a disjunctive nature that is not necessarily

linear, see [Achterberg \(2007a\)](#). In contrast, dual proof analysis yields exactly one linear constraint.

For further details on conflict analysis in MIP we refer to [Achterberg \(2007a\)](#) and [Witzig et al. \(2019a\)](#) and the references therein. In this paper, we will focus on dual proof analysis for MINLP.

### 2.3 Conflict Analysis in MINLP

Only few publications are dealing with infeasibility in MINLP. Most of the literature is restricted to certain classes of MINLPs, e.g., conic certificates for convex MINLPs ([Coey et al. 2020](#)) which has been proven to be very successful on *mixed integer second-order cone* (MISOCP) problems. Purely theoretical results for *mixed integer semidefinite programs* (MISDP) were recently published in [Kellner et al. \(2019\)](#). Both publications deal with MINLPs that are infeasible as a whole, and not with the analysis of infeasible subproblems to learn information during the solution process.

For MINLP algorithms that are based on solving LP relaxations, in particular, for OA- and ECP-based solvers, conflict analysis methods for MIP can be applied under certain conditions. To this end, let us first recap the idea of constructing an LP relaxation for an MINLP.

During the tree search, nonlinear functions are approximated by linear functions if they are violated by a relaxation solution. Let  $\tilde{x}$  be a relaxation solution with  $g_k(\tilde{x}) > 0$ . If  $g_k$  is convex, a so-called *gradient cut*

$$g_k(\tilde{x}) + \nabla g_k(\tilde{x})(x - \tilde{x}) \leq 0$$

is added. If  $g_k$  is nonconvex, convex underestimators are used to relax  $g_k$ . For products of two variables, these are the so-called McCormick underestimators ([McCormick 1976](#)). More general nonlinear functions are typically decomposed into functions of a single variable, for which explicit underestimators are known, and products of two variables. If such an underestimator itself is linear, it is added to the LP relaxation directly, otherwise a gradient cut for the underestimator is derived (e.g., [Vigerske and Gleixner 2018](#), [Belotti et al. 2013](#)). Note that gradient cuts derived from convex functions are globally valid, while underestimators for non-convex functions (and their gradient cuts) typically involve some local bounds and are hence not globally valid. They are locally valid at the node for which they were created and its descendants.

For a subproblem  $s$  during the tree search, let  $\mathcal{G}^s := \{l_1^s, \dots, l_q^s\}$  be the index set of all linear approximations of all  $g_k$  with  $k \in \mathcal{K}$  that have been added at the node corresponding to  $s$  or any of its ancestors. Hence, it is the current set of (local) *linear relaxation cuts*; all are valid at  $s$ . Let  $G^s$  be the matrix containing all of these linearizations and  $d^s$  be the corresponding right-hand sides. Thus, the LP relaxation solved for subproblem  $s$  reads as

$$\min\{c^\top x \mid Ax \leq b, G^s x \leq d^s, \ell' \leq x \leq u'\}. \quad (6)$$



We denote the set of linearizations added at the root node by  $\mathcal{G}^0$ . During (spatial) branch-and-bound the set of linearizations expands along each path of the tree: It holds that  $\mathcal{G}^0 \subseteq \mathcal{G}^{s_1} \subseteq \dots \subseteq \mathcal{G}^{s_p} \subseteq \mathcal{G}^s$  for each path  $(0, s_1, \dots, s_p, s)$ . In analogy to solving MIPs, if (6) is infeasible each ray  $(y, w, r^\ell, r^u)$  in its dual can be used to construct a proof of local infeasibility. Here,  $y_i$  are the dual variables corresponding to  $A_i$ ,  $w_l$  are the dual variables corresponding to  $G_l^s$  for all  $l \in \mathcal{G}^s$ , and  $r_j^\ell, r_j^u$  denote the reduced costs (the duals of the lower and upper bound constraints) of every variable  $x_j$ . Note that the reduced costs of  $x_j$  are given by  $c_j - y^\top A_{.j} - w^\top G_{.j}^s$ .

Hence, a local infeasibility proof with respect to the local bounds  $\ell'$  and  $u'$  is given by

$$y^\top b + w^\top d^s + \sum_{j \in \mathcal{N}} (r_j^u u'_j - r_j^\ell \ell'_j) < 0. \quad (7)$$

In contrast to (5) the constraint

$$y^\top Ax + w^\top G^s x \leq y^\top b + w^\top d^s \quad (8)$$

is not globally valid in general because linearizations of nonconvex constraints might rely on intermediate local bounds. Conflict graph and dual proof analysis as introduced in [Achterberg \(2007a\)](#) and [Witzig et al. \(2017, 2019a\)](#) only consider globally valid reasons of infeasibility. To this end, the locally valid constraint (8) is relaxed to a globally valid constraint, considering only the dual multipliers for the original linear system  $Ax \leq b$  and the globally valid linearization  $\mathcal{G}^0 \leq d^0$ :

$$y^\top Ax + \bar{w}^\top G^s x \leq y^\top b + \bar{w}^\top d^s, \quad (9)$$

where  $\bar{w}_l := w_l$ , if  $l \in \mathcal{G}^0$ , and  $\bar{w}_l := 0$ , otherwise. In general, this relaxed constraint might not prove infeasibility of subproblem  $s$ . If, however, (9) is a valid proof of local infeasibility, i.e., no locally valid linearization is needed to prove local infeasibility, we can use a relaxed infeasibility certificate

$$y^\top b + \bar{w}^\top d^s + \sum_{j \in \mathcal{N}} (\bar{r}_j^u u'_j - \bar{r}_j^\ell \ell'_j) < 0, \quad (10)$$

with  $\bar{r} := c_j - y^\top A_{.j} - \bar{w}^\top G_{.j}^s$ , to generate conflict constraints. In this case, (10) is an alternative infeasibility proof for subproblem  $s$ , no locally valid linearization is needed to prove local infeasibility, and all conflict analysis techniques known from MIP can be used straightforward. As reported in [Witzig et al. \(2019b\)](#), this situation could only be observed in 5% of all infeasible subproblems. This observation corresponds to the result reported in [Vigerske and Gleixner \(2018\)](#) that applying conflict analysis techniques as used for solving MIPs has almost no impact when solving MINLPs. In the vast majority of cases, at least one locally valid linearization cut is needed to prove local infeasibility, and MIP conflict analysis cannot be readily applied. This brings us to the idea of local dual proofs.

### 3 Local Dual Proofs for LP-based Branch-and-Bound

In MIP, both conflict graph analysis and dual proof analysis rely on globally valid proofs. In most MIP solvers, local cuts are applied rarely, if at all. This is very different for non-convex MINLP solvers which rely on local linearization cuts.

The crucial question is how to deal with linearization cuts that are only locally valid. Note that the same question might rise within a MIP solver using local cuts, e.g., FICO Xpress. Analogous to MINLP, local cuts in MIP are based on variable bounds that are not globally valid.

To incorporate local linearizations of nonlinear constraints we propose lifting infeasibility proofs as far as possible towards the root. This idea is motivated by two observations. First, an infeasibility proof at a subproblem  $s$  of an LP-based branch-and-bound does not necessarily need to contain linearization cuts of nonconvex constraints created at the current node. Second, LP infeasibility proofs are typically not minimal. Consequently, it might be possible to set some of the proof's coefficients to zero and still have a valid proof of infeasibility, see also (Achterberg 2007a). This brings us to the idea of performing a greedy search for a set  $\hat{\mathcal{G}}$  with  $\mathcal{G}^0 \subseteq \hat{\mathcal{G}} \subseteq \mathcal{G}^s$ , such that  $\hat{\mathcal{G}}$  gives rise to an infeasibility proof

$$y^\top b + \hat{w}^\top d^s + \sum_{j \in \mathcal{N}} (\hat{r}_j^u u'_j - \hat{r}_j^\ell \ell'_j) < 0 \quad (11)$$

of subproblem  $s$ . Here,  $\ell'$  and  $u'$  are the local bounds at node  $s$  and  $\hat{w}_l$  is defined as  $\hat{w}_l := w_l$ , if  $l \in \hat{\mathcal{G}}$ , and  $\hat{w}_l := 0$ , otherwise; consequently,  $\hat{r}$  reads as  $\hat{r}_j := c_j - y^\top A_{.j} - \hat{w}^\top G_{.j}^s$ . The goal of the greedy search is to minimize  $\hat{s} = 0, \dots, s$ , such that  $\hat{\mathcal{G}} \subseteq \mathcal{G}^{\hat{s}}$ .

It follows that the dual proof constraint derived from the infeasibility certificate (11) is valid for the search tree induced by subproblem  $\hat{s}$ . Hence, the infeasibility proof might be lifted to an ancestor  $\hat{s}$  of the subproblem  $s$  it was created for, if all local information used for the proof were already available at  $\hat{s}$ . In other words, node  $\hat{s}$  is the ancestor of  $s$  closest to the root for which we can reduce the infeasibility proof such that it is still a valid proof, but only uses globally valid linearizations plus local linearizations that have been created at  $\hat{s}$  or one of its ancestors (but no local linearizations from the nodes between  $\hat{s}$  and  $s$  or  $s$  itself).

Note that it would be possible to apply conflict graph analysis to (11), too. However, this would introduce a computational overhead because the order of locally applied bound changes and separated local linearizations needs to be tracked and maintained. Hence, we refrain from applying conflict graph analysis on locally valid infeasibility certificates in our implementation.

## 4 Nonlinear Dual Proofs for NLP-based Branch-and-Bound

Recall that next to LP/MIP-based branch-and-bound, another common method to solve MINLPs is NLP-based branch-and-bound. In this section, we will discuss theoretical considerations how conflict analysis can be directly applied to convex nonlinear relaxations of MINLPs. More precisely, we will describe how a generalization of LP dual proof analysis can be derived from the KKT conditions of convex NLPs. For ease of notation, we will assume in this section that the MINLP itself is convex, but all methodology can be generalized to convex relaxations of nonconvex MINLPs.

Given a convex MINLP

$$\min_{x \in X} \{f(x) \mid g_k(x) \leq 0 \forall k \in \mathcal{K}, h_e(x) = 0 \forall e \in \mathcal{E}, x_j \in \mathbb{Z} \forall j \in \mathcal{I}\}, \quad (12)$$

where  $f, g_k$  are convex, continuously differentiable functions over  $\mathbb{R}^n$  and  $h_e$  are affine functions. For every optimal solution  $x^*$  of (12) of the (convex) NLP relaxation of (12) there exists  $\lambda \geq 0$  such that it holds that

$$\nabla f(x^*) + \sum_{k \in \mathcal{K}} \lambda_k \nabla g_k(x^*) + \sum_{e \in \mathcal{E}} \mu_e \nabla h_e(x^*) = 0 \quad (13)$$

$$\lambda_k g_k(x^*) = 0. \quad (14)$$

These conditions originate from the so-called *Karush-Kuhn-Tucker-Conditions* (Kuhn and Tucker 1951). The left-hand side of Equality (13) is the gradient of the *Lagrangian function*

$$\mathcal{L}(x, \lambda, \mu) := f(x) + \sum_{k \in \mathcal{K}} \lambda_k g_k(x) + \sum_{e \in \mathcal{E}} \mu_e h_e(x), \quad (15)$$

with  $\lambda \geq 0$  and  $\mu \in \mathbb{R}^{|\mathcal{E}|}$ . The *Lagrangian dual function*  $q: \mathbb{R}^{|\mathcal{K}|} \times \mathbb{R}^{|\mathcal{E}|} \mapsto \mathbb{R}$  is defined as

$$q(\lambda, \mu) := \inf_{x \in X} \mathcal{L}(x, \lambda, \mu). \quad (16)$$

Further, the dual problem of (12) is given by maximizing the Lagrangian dual function

$$\sup_{\lambda \geq 0, \mu} q(\lambda, \mu) := \sup_{\lambda \geq 0, \mu} \inf_{x \in X} \mathcal{L}(x, \lambda, \mu) \quad (17)$$

and yields the tightest lower bound on the optimal value of (12). If (12) is infeasible a proof of infeasibility can be derived from the following nonlinear version of Farkas' Lemma, introduced in Avriel (2003, Theorem 3.2):

**Theorem 1** *Let  $g_k: \mathbb{R}^n \mapsto \mathbb{R}$  with  $k \in \mathcal{K}$  be convex, continuously differentiable functions over  $\mathbb{R}^n$ ,  $h_e: \mathbb{R}^n \mapsto \mathbb{R}$  with  $e \in \mathcal{E}$  be affine functions, and  $x^* \in X$ . Exactly one of the following systems will be satisfied.*

(a) There exists  $(\lambda^*, \mu^*) \in \mathbb{R}^{|\mathcal{K}| \times |\mathcal{E}|}$  with  $\lambda^* \geq 0$  such that

$$\begin{aligned} \nabla_x \mathcal{L}(x^*, \lambda^*, \mu^*) &= \nabla f(x^*) + \sum_{k \in \mathcal{K}} \lambda_k^* \nabla g_k(x^*) + \sum_{e \in \mathcal{E}} \mu_e^* \nabla h_e(x^*) = 0 \\ \lambda_k^* g_k(x^*) &= 0, \quad g_k(x^*) \leq 0 \quad \forall k \in \mathcal{K} \\ h_e(x^*) &= 0 \quad \forall e \in \mathcal{E}. \end{aligned}$$

(b) For all  $(\lambda^*, \mu^*) \in \mathbb{R}^{|\mathcal{K}| \times |\mathcal{E}|}$  with  $\lambda^* \geq 0$  such that

$$\mathcal{L}(x, \lambda^*, \mu^*) = f(x) + \sum_{k \in \mathcal{K}} \lambda_k^* g_k(x) + \sum_{e \in \mathcal{E}} \mu_e^* h_e(x) > 0 \quad (18)$$

holds for all  $x \in X$ .

In other words, the (primal) NLP relaxation of MINLP (12) is infeasible if and only if there exists an infinite direction  $(\lambda^*, \mu^*)$  in the dual. Thus,  $(\lambda^*, \mu^*)$  proves infeasibility of MINLP (12) and consequently

$$\sum_{k \in \mathcal{K}} \lambda_k^* g_k(x) + \sum_{e \in \mathcal{E}} \mu_e^* h_e(x) \leq 0 \quad (19)$$

is a valid inequality for (12). It is a conic combination of the inequality constraints  $g_k(x) \leq 0$  plus a linear combination of the equations  $h_e(x) = 0$ . Inequality (19) is the convex optimization analogue of the dual proof constraint (4).

Assume that constraint (19) is given as proof of infeasibility for a subproblem within an NLP-based branch-and-bound. If no local cuts are involved in the infeasibility proof, inequality (19) is a globally valid convex nonlinear constraint. Note in this context that gradient cuts are globally valid.

Clearly, inequality (19) holds for all non-negative  $\lambda^*$ . The following observation makes the concrete  $(\lambda^*, \mu^*)$  from the nonlinear infeasibility proof interesting to use as global information inside a branch-and-bound tree search for convex MINLP. Consider the linearization at an infeasible point  $x^* \in X$

$$g_k(x^*) + \nabla g_k(x^*)^\top (x - x^*) \leq 0 \quad \Leftrightarrow \quad \nabla g_k(x^*)^\top x \leq \nabla g_k(x^*)^\top x^* - g_k(x^*) \quad \forall k \in \mathcal{K}. \quad (20)$$

Then, the corresponding dual multipliers  $\lambda^*, \mu^*$  give the linear Farkas system

$$\begin{aligned} \sum_{k \in \mathcal{K}} \lambda_k^* \nabla g_k(x^*) + \sum_{e \in \mathcal{E}} \mu_e^* \nabla h_e(x^*) &= 0 \\ \sum_{k \in \mathcal{K}} \lambda_k^* (\nabla g_k(x^*)^\top x^* - g_k(x^*)) + \sum_{e \in \mathcal{E}} \mu_e^* \nabla h_e(x^*)^\top x^* &< 0. \end{aligned} \quad (21)$$

Thus, for every infeasible convex MINLP an LP relaxation can be constructed from which a linear proof of infeasibility can be derived. This could be done by linearizing all nonlinear constraints  $g_k$  with  $\lambda_k > 0$  at an (arbitrary) infeasible

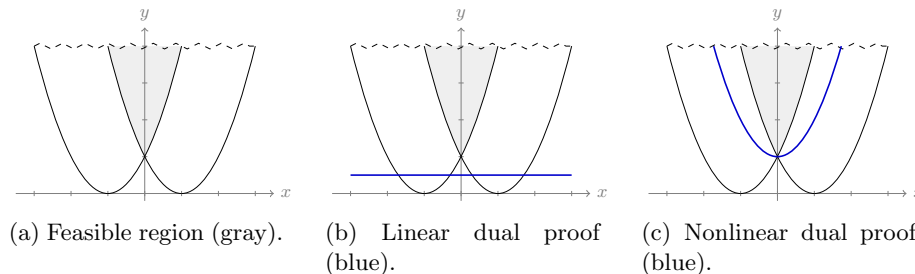


Figure 1: Illustration of linear and nonlinear dual proofs of Example 2 projected into the  $y$ -space.

point. However, using a linear dual proof constraint derived from (21) will in general be weaker than the nonlinear dual proof (19). An illustration of the strength of linear and nonlinear underestimators is given in the following example and Figure 1.

**Example 2** Consider the following convex nonlinear system.

$$\begin{aligned} g_1(x, y) : & \quad (x - 1)^2 - y \leq 0 \\ g_2(x, y) : & \quad (x + 1)^2 - y \leq 0 \end{aligned}$$

with  $x, y \geq 0$ , see Figure 1a. Assume we perform a branching  $y \leq \frac{1}{2}$ , thereby rendering the system infeasible by a local bound change. Then,  $\lambda_1 = \lambda_2 = \frac{1}{2}$ , together with  $\lambda_3 = 1$  for the branching constraint, are valid dual multipliers proving infeasibility. Linearizing  $g_1$  and  $g_2$  at the infeasible point  $(0, \frac{1}{2})$  gives

$$2x - y + \frac{1}{2} \leq 0 \quad \text{and} \quad -2x - y + \frac{1}{2} \leq 0.$$

Aggregating both linearizations, see Figure 1b, with  $\lambda_1, \lambda_2$  yields  $-y + \frac{1}{2} \leq 0$  as a linear dual proof constraint. The linear dual proof constraints proves infeasibility for  $y < 0.5$ . In contrast to that, aggregating both nonlinear constraints  $g_1$  and  $g_2$  yields the nonlinear dual proof constraint  $x^2 + 1 - y \leq 0$ , see Figure 1c. The nonlinear dual proof constraint proves infeasibility for  $y < x^2 + 1$ , which is stronger than the linear proof.

As in the case of dual proof analysis for MIP, inequality (19) is a single inequality that would have directly provided an infeasibility proof. If (19) had been part of the constraint set, the relaxation would not have to be solved for the current subproblem; instead, infeasibility could have been proven by domain propagation.

The hope (which is true for MIP) is that (19) is a good candidate to detect infeasibility by propagation (under the use of integrality information) in other parts of the search tree, and might be a meaningful aggregation of problem

constraints to create cuts from. As in the MIP case, infeasibility information might be used in other contexts, consider hybrid branching (Achterberg and Berthold 2009), conflict-driven diving heuristics (Witzig and Gleixner 2019), and also rapid learning (Berthold et al. 2010, 2019).

For many NLP solvers, in particular dual active set methods (Murty and Yu 1988, Nocedal and Wright 2006, Forsgren et al. 2016) and barrier algorithms (Mehrotra 1992, Mészáros 1999, Wächter and Biegler 2006), dual multipliers will be readily available. The added advantage of active set methods is that they typically yield a sparse dual weight-vector  $(\lambda, \mu)$ .

## 5 Nonlinear Dual Proofs: Implementation Aspects

In this section, we introduce a dual multiplier filtering heuristic, which proved to be an important details of our implementation, and we discuss the impact of different types of NLP solvers on the presented algorithms.

### 5.1 Dual Multiplier Filtering Heuristic

Let  $(\lambda^*, \mu^*) \in \mathbb{R}_{\geq 0}^{|\mathcal{K}|} \times \mathbb{R}^{|\mathcal{E}|}$  be a set of dual multipliers proving infeasibility of (12). The support of  $(\lambda^*, \mu^*)$  is not minimal in general; Every set of dual multipliers  $(\bar{\lambda}, \bar{\mu}) \in \mathbb{R}_{\geq 0}^{|\mathcal{K}|} \times \mathbb{R}^{|\mathcal{E}|}$  with  $\text{supp}(\lambda^*) \subseteq \text{supp}(\bar{\lambda})$  and  $\text{supp}(\mu^*) \subseteq \text{supp}(\bar{\mu})$  satisfying (18) yields a (nonlinear) dual proof. Hence, we might be able to obtain a dual proof involving less of the original problem constraints simply by setting some of the dual multipliers to zero. In particular, in some cases it might be possible to remove entire classes of constraints in this fashion.

In our implementation we always try to prefer linear dual proofs over nonlinear ones. One reason for preferring linear constraints is the fast propagation compared to nonlinear constraints in general, i.e., linear time in the number of nonzero variables compared to linear time in the number of expressions. Moreover, quadratic constraints are preferred over general nonlinear. Therefore, we subsequently check whether the following constraints prove infeasibility:

1.  $\sum_{e \in \mathcal{E}} \mu_e^* h_e(x) \leq 0$
2.  $\sum_{i \in \mathcal{Q}} \lambda_i^* g_i(x) + \sum_{e \in \mathcal{E}} \mu_e^* h_e(x) \leq 0$ , where  $\mathcal{Q} := \{i \in \mathcal{K} \mid g_i \text{ is quadratic}\}$
3.  $\sum_{i \in \mathcal{K}} \lambda_i^* g_i(x) + \sum_{e \in \mathcal{E}} \mu_e^* h_e(x) \leq 0$

and immediately stop if we found a valid proof.

Analogous to linear dual proof constraints, nonlinear dual proof constraints are used only for propagation within our implementation.

## 5.2 Interior Point vs. Active Set Methods

The preferred technique for solving the relaxations within an LP-based (spatial) branch-and-bound is the dual Simplex algorithm (Lemke 1954). Simplex-based LP solvers typically provide the dual multipliers proving local infeasibility. However, the set of dual multipliers returned by the Simplex algorithm is not minimal in general. Note, calculating a maximally sparse vector of dual multipliers that is sufficient to render infeasibility is strongly  $\mathcal{NP}$ -hard (Chakravarti 1994). Such a set of multipliers is well-known as *minimal irreducible infeasible set* (minIIS) (e.g., Chinneck and Dravnieks 1991, Gleeson and Ryan 1990).

Similar to LP solvers, dual multipliers are readily available in many NLP solvers. This holds in particular for *dual active set methods* (e.g., Murty and Yu 1988, Nocedal and Wright 2006, Forsgren et al. 2016) and barrier algorithms (e.g., Mehrotra 1992, Mészáros 1999, Wächter and Biegler 2006). A major advantage of active set methods is that they typically yield a sparse dual weight-vector  $(\lambda, \mu)$ . Like in the linear case, the problem is that the initial reason will typically be too large to be meaningful. Intuitively, one would expect that a sparse vector of dual multipliers might lead to nonlinear dual proofs with a smaller support. Of course, this intuition is not true in general but it is worthwhile to investigate whether active set methods lead to nonlinear dual proofs that are more effective than those generated with a dense vector of multipliers, like it is the case for barrier algorithms. We support both types of NLP solvers within our implementation and compare their impact on dual proof analysis as part of our computational study.

## 6 Computational Results

In our computational study, we address three questions:

- When solving general MINLPs with an LP-based branch-and-bound, what is the impact of using a combined approach of applying (global) dual proof analysis in addition to conflict graph analysis?
- Given that local linearization cuts often prohibit finding globally valid proof constraints: What is the impact of our suggested strategy to generate local dual proofs when solving nonconvex MINLPs with an LP-based branch-and-bound?
- When solving general MINLPs with an NLP-based branch-and-bound approach, what is the impact of our newly proposed nonlinear dual proofs?

In order to answer these three questions, we carried out an extensive computational study. In the first part of this section, we analyze the impact of LP-based conflict analysis in SCIP within an LP-based branch-and-bound. To do so, we compare SCIP without LP-based conflict analysis (`nolpinf`) against SCIP using conflict graph analysis solely (`confgraph`), and SCIP using a combined approach of conflict graph analysis and dual proof analysis (`combined`).

In the second part of this section, we compare `nolpinf` with `combined` and an extension of `combined` that exploits locally valid dual proof constraints. We refer to the latter as `combined-local`. Note again that for convex MINLPs, all proof constraints are globally valid. Consequently, this part of the study is only conducted for nonconvex MINLPs.

In the last part of this section, we present computational results on the impact of nonlinear dual proof constraints. Based on an NLP-based branch-and-bound, we compare `SCIP` without conflict analysis (`noconflict`) to `SCIP` applying conflict graph analysis to infeasibilities derived during constraint propagation (`nonlpinf`). Further, we compare both settings to `SCIP` deriving nonlinear dual proof constraints from infeasible convex NLP relaxations (`nlpinf`).

All experiments were performed with the academic solver `SCIP` (git hash `fd3e45275d`, based on `SCIP 6.0.2`) (Gleixner et al. 2018). For the results presented in Sections 6.1 and 6.2, we used `SoPlex 4.0.0` as LP solver. For the results presented in Section 6.3, we used `Ipopt 3.12.11` (Wächter and Biegler 2006) and `FilterSQP` (Fletcher and Leyffer 1998) as NLP solvers.

To evaluate the generated data, we used the *interactive performance evaluation tool* (IPET)<sup>8</sup>. We ran the experiments on a cluster of identical machines equipped with Intel Xeon E5-2690 CPUs with 2.6 GHz and 128 GB of RAM; a time limit of 7200 seconds was set. To account for the effect of performance variability (Lodi and Tramontani 2013) all experiments were performed with three different constraint permutations. We used `MINLPLIB`<sup>9</sup> as a test set, excluding instances that are purely continuous, i.e., do not contain any integer variables. This test set consists of 1029 publicly available MINLP problems. Every combination of MINLP problem and permutation is treated as an individual observation, effectively resulting in a test set of 3087 instances. We will use the term “instance” when referring to a problem-permutation combination.

All tables shown in the remainder of this section contain aggregated results. Detailed results with instance-wise computational results can be found in a GitHub repository<sup>10</sup>. Note that for every table we excluded instances that either could be solved at the root node by all settings or for which at least one setting finished with numerical violations.

The tables in this paper distinguish between instances for which every considered setting hit the time limit (`timelimit`) and instances that could be solved by at least one considered setting (`1-opt`). The set `1-opt` is further divided into affected instances and into a hierarchy of increasingly harder classes “ $\geq k$ ”. We call an instance affected when it did not take the same solution path for all settings of the respective table. Class “ $\geq k$ ” contains all affected instances for which at least one setting needed at least  $k$  seconds. As explained by Achterberg and Wunderling (2013), this excludes instances that are “easy” for all settings in an unbiased manner. Moreover, if both convex and nonconvex MINLPs are considered, the respective table distinguishes between convex and nonconvex affected instances.

<sup>8</sup><https://github.com/GregorCH/ipet>

<sup>9</sup><http://www.minlplib.org/>, git hash `a71254dc`

<sup>10</sup><https://github.com/jakobwitzig/conflict-analysis-minlp>



For every group of instances the tables show:

- the number of instances in the respective group (2<sup>nd</sup> column: #),
- the number of solved instances (**S**)
- the shifted geometric mean of run time, where a shift of 1 was used (**T**)
- the relative run time with respect to the baseline setting (**T<sub>Q</sub>**)
- the shifted geometric mean of tree nodes, where a shift of 100 was used (**N**)
- the relative number of tree nodes with respect to the baseline setting (**N<sub>Q</sub>**)
- the dual integral with respect to the best know primal solution (**DI**)
- the relative dual integral with respect to the baseline setting (**DI<sub>Q</sub>**)

Relative solving times of setting  $s$  are defined by the quotient  $t_s/t_b$ , where  $t_s$  is the mean solving time of setting  $s$  and  $t_b$  is the mean solving time of setting  $b$  that is used as a baseline. An analogous definition holds for explored branch-and-bound nodes and the dual integral. Thus, a number less than one implies that the setting  $s$  is superior and a number greater than one implies that it is inferior to the baseline setting  $b$ . In the following tables, an improvement or deterioration by at least 5% is highlighted in bold and blue or italic and red, respectively.

The (relative) number of nodes is computed over the set of instances that could be solved by all settings. For the group 1-opt, 1376 out of 1403 (Table 1), 606 out of 632 (Table 2), and 598 out of 626 (Table 3) fulfilled this criterion.

## 6.1 LP-based Conflict Analysis

In this section, we address our first question of what the impact of LP-based conflict analysis is when solving MINLPs with an LP-based branch-and-bound. Table 1 summarizes our findings. It compares three different settings: `nolpinf` for a version of `SCIP` that does not use LP-based conflict analysis, `confgraph` for a version of `SCIP` that applies conflict graph analysis to infeasible LP relaxations, and `combined` for a version of `SCIP` that uses dual proof analysis in addition to conflict graph analysis for infeasible LPs. The setting `confgraph` is the default setting of `SCIP` and corresponds to what has been used by [Vigerske and Gleixner \(2018\)](#), which to the best of our knowledge is the only computational study of conflict analysis techniques for MINLP so far.

Our results confirm the findings of the named paper that the impact of conflict graph analysis on the overall running time is marginal, yet slightly positive. Not surprisingly, the technique works better on the subset of convex MINLPs, a distinction the authors did not consider. Our results indicate that applying dual proof analysis in addition to conflict graph analysis is beneficial.

Table 1: Impact on LP-based branch-and-bound when using a combined approach of dual proof and conflict graph analysis: aggregated computational results on MINLPLIB.

	#	nolpinf				confgraph				combined			
		S	T	N	DI	S	T <sub>Q</sub>	N <sub>Q</sub>	DI <sub>Q</sub>	S	T <sub>Q</sub>	N <sub>Q</sub>	DI <sub>Q</sub>
1-opt	1403	1387	15.64	1463	895	1386	0.991	1.001	0.952	1394	0.970	0.963	0.954
timeout	1022	0	7200.00	-	231626	0	1.000	-	0.959	0	1.000	-	<b>0.929</b>
affected	908	892	31.04	4431	1606	891	0.988	1.002	<b>0.930</b>	899	0.956	<b>0.945</b>	<b>0.933</b>
$\geq 10$	553	537	148.40	22441	4743	536	0.979	0.975	<b>0.893</b>	544	<b>0.933</b>	<b>0.911</b>	<b>0.899</b>
$\geq 100$	295	279	646.17	129365	22502	278	0.981	0.991	<b>0.842</b>	286	<b>0.920</b>	<b>0.936</b>	<b>0.852</b>
$\geq 1000$	131	115	2173.90	422656	75451	114	1.003	1.005	<b>0.746</b>	122	<b>0.910</b>	0.968	<b>0.804</b>
convex	485	483	17.67	3834	1269	483	0.951	0.984	<b>0.855</b>	483	<b>0.917</b>	<b>0.913</b>	<b>0.867</b>
nonconvex	423	409	58.52	5269	2096	408	1.030	1.025	1.020	416	0.999	0.984	1.012

We observed an overall speedup of 3%, with speedups of 8% and more on instances that take more than 100 seconds, more than 1000 seconds to solve, and on the subset of convex MINLPs. This goes hand in hand with a significant reduction of the average search tree size and an improvement in the dual integral of more than 10% for many subsets that we considered. Notably, also for the set of instances that timed out with all settings, we observed an improvement in the dual integral by about 7%.

Similar to what is known from solving MIPs, our results indicate that conflict analysis for MINLPs works especially well for infeasible instances and feasibility instances<sup>11</sup>, i.e., for instances where the tree is not pruned by bounding with respect to an incumbent solution. For such instances, we observed an above average speedup of 24% (**confgraph**) and 38% (**combined**) compared to **noconflict**. However, this group of instances forms a rather small part of our test set (33 in 1-opt, 14 affected).

As expected, for nonconvex MINLPs, the technique is almost performance-neutral. Globally valid proofs could rarely be constructed, therefore the benefit on a few instances is compromised by a slight computational overhead on many instances. It is worth mentioning that we could solve eight more nonconvex MINLPs when using dual proof analysis.

## 6.2 LP-based Local Dual Proofs

In this section, we address our second question of what the impact of generating local dual proofs is when solving nonconvex MINLPs with an LP-based branch-and-bound. Table 2 depicts our results. Columns **nolpinf** and **combined** take the same meaning as before, Column **combined-local** shows the performance of a version that generates both, conflict constraints and dual proof constraints that are allowed to be locally valid. We see that for harder instances, the impact of standard conflict analysis is even slightly negative. In marked contrast, employing local dual proofs for nonconvex MINLPs led to a performance im-

<sup>11</sup>We call an instance a feasibility instance, if the optimal value of its relaxation matches the optimal solution value

Table 2: Impact of using local dual proof within LP-based branch-and-bound: aggregated computational results on the subset of nonconvex instances of MINLPLIB.

	#	nolpinf				combined				combined-local			
		S	T	N	DI	S	T <sub>Q</sub>	N <sub>Q</sub>	DI <sub>Q</sub>	S	T <sub>Q</sub>	N <sub>Q</sub>	DI <sub>Q</sub>
1-opt	632	614	31.62	2123	1150	621	0.998	0.985	1.009	623	0.975	0.964	1.004
timeout	862	0	7200.00	-	226311	0	1.000	-	<b>0.923</b>	0	1.000	-	<b>0.932</b>
affected	441	423	58.63	5227	2065	430	0.997	0.979	1.012	432	0.965	<b>0.949</b>	1.006
≥ 10	309	291	191.87	15244	5884	298	1.005	0.999	1.021	300	0.959	0.954	1.013
≥ 100	179	161	736.74	68520	22734	168	1.045	<i>1.113</i>	1.047	170	0.963	1.030	1.034
≥ 1000	80	62	2593.38	167820	74290	69	1.018	<i>1.163</i>	1.036	71	<b>0.877</b>	1.013	0.965

provement with respect to the overall running time on all subset of instances that we considered. The overall impact was about 2.5% speedup. For instances that need more than 1000 seconds to solve, the difference was most pronounced, but given the small size of this subset, the result should be taken with a grain of salt. The slight deterioration in the number of nodes is due to a single outlier instance which creates a significantly larger search tree while taking only twice as much run time.

The impact on the number of branch-and-bound nodes is roughly similar to the impact on running time. The dual integral, however, seems to be hardly affected. This can partially be explained by this technique being local and thereby only affecting the part of the tree that is currently being processed. In classical conflict analysis, the generation of new globally valid constraints might, e.g., trigger an immediate reduction of global variable bounds by propagating the new constraints, which in turn might raise the global (not only the local) dual bound. It should be said, though, that on the set of instances that timed out with all settings, we did observe an improvement for the dual integral. Overall, the results give a strong indication that the generation of local infeasibility proofs for nonconvex MINLPs is beneficial.

Figure 2 visualizes the effect of lifting locally valid dual proofs, in an instance-wise manner. For every nonconvex MINLP in our test set where at least 50 infeasible LP relaxations were encountered (1142 instances) we analyzed the reduction of depth at which the dual proof is valid. More precisely, for each instance we calculated the following three percentages:

- The portion of dual proofs for which at least one linearization cut that is only valid at the current node is needed to prove infeasibility (group "local"). For such proofs, no depth reduction is possible and they have to be discarded.
- The portion of dual proofs for which no locally valid linearization cut is needed (group "global"). These proofs are globally valid and can be lifted to the root node. Hence, they can be used for propagation throughout the remainder of the search.
- The portion of dual proofs for which at least one local linearization cut is needed, but none of the current node. These proofs are locally valid at

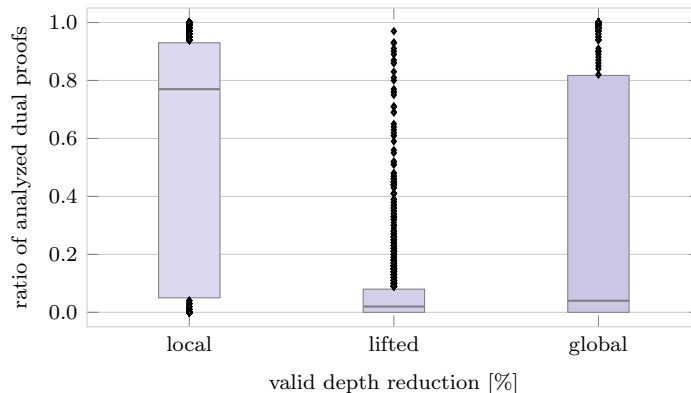


Figure 2: Instance-wise distribution dual proof constraints that are only locally valid at the node where they have been created, constraints that can be lifted to an inner node of the search tree, and constraints that are globally valid.

some ancestor of the current node, but not at the root. They can be used for searching parts of the tree which share that ancestor and are therefore added as locally valid constraints to the named ancestor node. (group "lifted").

For each group the figure shows a box-plot (McGill et al. 1978), where the box ranges indicates the second and the third quartile. The median is marked with a bold, horizontal solid line, each instance outside the box is indicated by a dot.

The left box plot shows that it is a common case that most proofs are only locally valid and, therefore, have to be discarded. The median bar is at 77%, which means that for half of the considered instances, we could not lift the generated dual proof for at least 77% of all analyzed infeasible LPs. For 25% of the instances, this share was even larger than 93%. Note, however, that the 75%-percentile is comparably low, at 5%.

The right box plot shows that the next common case is proofs being globally valid. For 25% of the instances, more than 82% of the proofs could be lifted to the root node. The median is at 4%, which means that for 50% of the instances, at least 4% of the proofs could be lifted to the root. On 35% of the instances, we observed that no globally valid proof could be derived at all.

The middle box plot finally demonstrates two things. Firstly, it shows that some form of lifting to an inner node is possible in a majority of instances, though typically not for a large share of proofs. For half of the instances, at least 2% of the proofs could be lifted to an inner node and for 25% of the instances, at least 8% of the proofs could be lifted. Secondly, we observe that the outlier dots are all over the place. There are various instances for which a large share of dual proofs can be lifted to inner nodes.

Table 3: Impact of nonlinear dual proofs on NLP-based branch-and-bound: aggregated results on MINLPLIB.

	#	noconflict				nonlpinf				nlpinf			
		S	T	N	DI	S	T <sub>Q</sub>	N <sub>Q</sub>	DI <sub>Q</sub>	S	T <sub>Q</sub>	N <sub>Q</sub>	DI <sub>Q</sub>
1-opt	626	601	66.02	505	1303	608	<b>0.937</b>	<b>0.948</b>	<b>0.950</b>	623	<b>0.766</b>	<b>0.887</b>	<b>0.877</b>
timeout	1660	0	7200.00	-	324698	0	1.000	-	<b>0.916</b>	0	1.000	-	<b>0.910</b>
affected	229	205	286.84	1529	3807	211	<b>0.839</b>	<b>0.870</b>	<b>0.875</b>	226	<b>0.488</b>	<b>0.729</b>	<b>0.712</b>
≥ 10	210	186	426.86	1987	4899	192	<b>0.827</b>	<b>0.860</b>	<b>0.866</b>	207	<b>0.467</b>	<b>0.716</b>	<b>0.693</b>
≥ 100	160	136	1018.40	3842	6909	142	<b>0.789</b>	<b>0.826</b>	<b>0.834</b>	157	<b>0.466</b>	<b>0.708</b>	<b>0.644</b>
≥ 1000	78	54	3197.11	9095	11166	60	<b>0.759</b>	<b>0.852</b>	<b>0.771</b>	75	<b>0.404</b>	<b>0.741</b>	<b>0.478</b>
convex	151	135	240.21	1132	5864	135	0.995	0.999	1.003	151	<b>0.453</b>	<b>0.806</b>	<b>0.751</b>
nonconvex	78	70	404.29	2756	1622	76	<b>0.604</b>	<b>0.664</b>	<b>0.659</b>	75	<b>0.563</b>	<b>0.599</b>	<b>0.633</b>

### 6.3 NLP-based Conflict Analysis

In this section, we address our third and final question of what the impact of generating nonlinear dual proofs is when solving MINLPs with an NLP-based branch-and-bound. Table 3 presents, to the best of our knowledge, the first computational results for applying conflict analysis within an NLP-based branch-and-bound algorithm. We show three different settings: `noconflict` for a version of `SCIP` that does not use any form of conflict analysis, `nonlpinf` for a version of `SCIP` that applies conflict graph analysis to infeasibilities found by domain propagation, and `nlpinf` for a version of `SCIP` that in addition uses dual proof analysis for infeasible NLPs.

For the experiments in this section, we extended `SCIP` to work as an NLP-based branch-and-bound solver, using `Ipopt` as NLP solver. In our implementation, we always solve a convex NLP relaxation of the possibly non-convex MINLP. The convex NLP relaxation consists of all linear constraints, non-linear constraints that are recognized by `SCIP` to be convex, and linear underestimators of non-convex constraints. Due to the plugin-based design of `SCIP`, a lot of the functionality could easily be inherited from the LP-based implementation. For example, the NLP-based solver features a fully-fledged presolving algorithm, advanced domain propagation/bound tightening procedures and involved branching rules based on pseudo-costs and other history information. However, some functionality is still missing in this prototype implementation, e.g., the separation of combinatorial cutting planes, or an NLP-equivalent to reduced cost fixing. Furthermore, while the LP-based branch-and-bound has undergone years of solver parameter fine-tuning, this does not hold true for our new NLP-based branch-and-bound implementation. Consequently, it can be expected that the results are somewhat more pronounced than for the LP-based case.

Nevertheless, the extent of this distinction came as a surprise to us. First of all, we see that conflict graph analysis from infeasibilities found in domain propagation already has a significant impact. It leads to an overall speedup of more than 6% and shows consistent improvements throughout the board for all our performance measures. The generation of nonlinear dual proofs from infeasible NLP relaxations excels. It reduces the overall running time by more

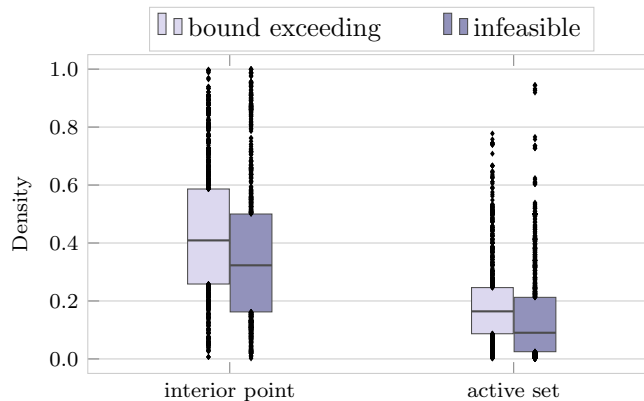


Figure 3: Instance-wise distribution of the density of constraints contributing to a dual proof constraint.

than 20%, cutting it down to less than half on the set of affected instances.

The results for the number of nodes and the dual integral are not as extreme, but still more than 20% in all subsets. On the set of instances where all methods timed out, the dual integral improved by 9%. Altogether, the results give clear evidence that the generation of nonlinear dual proofs helps to improve the performance of such algorithms.

1

In our implementation, we prefer linear infeasibility proofs derived from the nonlinear relaxation over quadratic infeasibility proofs and quadratic over general nonlinear infeasibility proofs, as described in Section 5.1. The set of instances that are affected by this dual multiplier filtering heuristic is comparably small, containing 75 instances. On this set of affected instances, however, we observe a clear performance improvement caused by the named heuristic. All three performance measures, the running time, the number of nodes and the dual integral, reduce significantly by 8%, by 2% and by 14%, respectively.

Finally, we made an additional experiment to investigate the impact of different NLP solving algorithms on the density of dual proofs as described in Section 5.2. The box plots in Figure 3 show that, as expected, the dual solutions generated by the active set method are much sparser than the ones from the interior point solver. When using an interior point solver, the average density of the dual solutions is larger than 40% for half of the instances for bound-exceeding proofs and larger than 32% for half of the instances for infeasibility proofs. When using an active set method, the median was at 16% and 9%, respectively. Consequently, the speedup achieved by dual proof analysis was slightly larger when using an active set method, but only in the range of one to two percent.

## 7 Conclusion

In this paper, we studied different ways to learn from infeasible subproblems when solving MINLPs. More precisely, we studied the impact of two different techniques – conflict graph analysis and dual proof analysis – and two different solution paradigms – LP-based branch-and-bound and NLP-based branch-and-bound – for convex and nonconvex MINLPs. Conflict graph analysis for MINLP has been previously described in the literature and shown to be of little impact. Although the generalization of dual proof analysis to LP-based branch-and-bound for MINLP is straight-forward, it has, to the best of our knowledge, not been studied before. We introduced a novel technique, lifting of local dual proofs for nonconvex MINLPs, and showed that a combination of conflict graph analysis and dual proof analysis lead to a speedup of more than 8% for convex MINLPs and 4.5% for nonconvex MINLPs in a state-of-the-art LP-based branch-and-bound solver.

The other main contribution of the paper is a generalization of dual proof analysis to NLP-based branch-and-bound for MINLP. We showed how to derive globally valid nonlinear constraints from the dual solution of a convex NLP relaxation of a local subproblem, using a variant of Farkas’ lemma based on KKT conditions. When implementing this technique within an NLP-based branch-and-bound solver, we observed a speedup of more than 20%. Our results clearly indicate that the analysis of infeasible subproblems plays an important role for the performance of global MINLP solvers and constitutes an interesting field of research.

**Acknowledgments** The work for this article has been conducted within the Research Campus Modal funded by the German Federal Ministry of Education and Research (fund number 05M14ZAM). We thank Stefan Vigerske and Ambros Gleixner for their helpful comments on an earlier version of the paper.

## References

- Achterberg T (2007a) Conflict analysis in mixed integer programming. *Discrete Optimization* 4(1):4–20.
- Achterberg T (2007b) *Constraint integer programming*. phdthesis, Technische Universität Berlin.
- Achterberg T, Berthold T (2009) Hybrid branching. van Hoesel WJ, Hooker JN, eds., *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, 6th International Conference, CPAIOR 2009*, volume 5547 of *Lecture Notes in Computer Science*, 309–311 (Springer Berlin Heidelberg).
- Achterberg T, Wunderling R (2013) Mixed integer programming: Analyzing 12 years of progress. Jünger M, Reinelt G, eds., *Facets of combinatorial optimization*, 449–481 (Springer Berlin Heidelberg), URL [http://dx.doi.org/10.1007/978-3-642-38189-8\\_18](http://dx.doi.org/10.1007/978-3-642-38189-8_18).

- Avriel M (2003) *Nonlinear programming: analysis and methods* (Courier Corporation).
- Belotti P, Berthold T, Neves K (2016) Algorithms for discrete nonlinear optimization in FICO Xpress. *2016 IEEE Sensor Array and Multichannel Signal Processing Workshop (SAM)*, 1–5 (IEEE).
- Belotti P, Kirches C, Leyffer S, Linderoth J, Luedtke J, Mahajan A (2013) Mixed-integer nonlinear optimization. *Acta Numerica* 22:1–131, URL <http://dx.doi.org/10.1017/S0962492913000032>.
- Belotti P, Lee J, Liberti L, Margot F, Wächter A (2009) Branching and bounds tightening techniques for non-convex MINLP. *Optimization Methods & Software* 24(4-5):597–634.
- Benders JF (1962) Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik* 4(1):238–252.
- Berthold T, Feydy T, Stuckey PJ (2010) Rapid learning for binary programs. Lodi A, Milano M, Toth P, eds., *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, 7th International Conference, CPAIOR 2010*, volume 6140 of *LNCS*, 51–55 (Springer Berlin Heidelberg).
- Berthold T, Stuckey PJ, Witzig J (2019) Local Rapid Learning for Integer Programs. Rousseau LM, Stergiou K, eds., *Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, 67–83 (Cham: Springer International Publishing), ISBN 978-3-030-19212-9, URL [http://dx.doi.org/https://doi.org/10.1007/978-3-030-19212-9\\_5](http://dx.doi.org/https://doi.org/10.1007/978-3-030-19212-9_5).
- Biegler LT, Grossmann IE, Westerberg AW (1997) Systematic methods for chemical process design .
- Bixby RE (2002) Solving real-world linear programs: A decade and more of progress. *Operations Research* 50(1):3–15.
- Bonami P, Biegler LT, Conn AR, Cornuéjols G, Grossmann IE, Laird CD, Lee J, Lodi A, Margot F, Sawaya N, et al. (2008) An algorithmic framework for convex mixed integer nonlinear programs. *Discrete Optimization* 5(2):186–204.
- Chakravarti N (1994) Some results concerning post-infeasibility analysis. *European Journal of Operational Research* 73(1):139 – 143, ISSN 0377-2217, URL [http://dx.doi.org/https://doi.org/10.1016/0377-2217\(94\)90152-X](http://dx.doi.org/https://doi.org/10.1016/0377-2217(94)90152-X).
- Chinneck JW, Dravnieks EW (1991) Locating minimal infeasible constraint sets in linear programs. *ORSA Journal on Computing* 3(2):157–168, URL <http://dx.doi.org/10.1287/ijoc.3.2.157>.
- Coey C, Lubin M, Vielma JP (2020) Outer approximation with conic certificates for mixed-integer convex problems. *Mathematical Programming Computation* 12(2):249–293, URL <http://dx.doi.org/10.1007/s12532-020-00178-3>.
- Cornuéjols G, Tütüncü R (2006) *Optimization methods in finance*, volume 5 (Cambridge University Press), ISBN 9781139460569.
- Dakin RJ (1965) A tree-search algorithm for mixed integer programming problems. *The Computer Journal* 8(3):250–255.
- Davey B, Boland N, Stuckey PJ (2002) Efficient intelligent backtracking using linear programming. *INFORMS Journal of Computing* 14(4):373–386.
- Duran MA, Grossmann IE (1986) An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical programming* 36(3):307–339.



- Fletcher R, Leyffer S (1994) Solving mixed integer nonlinear programs by outer approximation. *Mathematical programming* 66(1-3):327–349.
- Fletcher R, Leyffer S (1998) User manual for filtersqp. *Numerical Analysis Report NA/181, Department of Mathematics, University of Dundee, Dundee, Scotland* 35.
- Floudas CA (2000) Global optimization in design and control of chemical process systems. *Journal of Process Control* 10(2):125 – 134, ISSN 0959-1524, URL [http://dx.doi.org/10.1016/S0959-1524\(99\)00019-0](http://dx.doi.org/10.1016/S0959-1524(99)00019-0).
- Forsgren A, Gill PE, Wong E (2016) Primal and dual active-set methods for convex quadratic programming. *Mathematical programming* 159(1-2):469–508.
- Ginsberg ML (1993) Dynamic backtracking. *Journal of Artificial Intelligence Research* 1:25–46.
- Gleeson J, Ryan J (1990) Identifying minimally infeasible subsystems of inequalities. *ORSA Journal on Computing* 2(1):61–63, URL <http://dx.doi.org/10.1287/ijoc.2.1.61>.
- Gleixner A, Bastubbe M, Eifler L, Gally T, Gamrath G, Gottwald RL, Hendel G, Hojny C, Koch T, Lübbecke ME, Maher SJ, Miltenberger M, Müller B, Pfetsch ME, Puchert C, Rehfeldt D, Schlösser F, Schubert C, Serrano F, Shinano Y, Viernickel JM, Walter M, Wegscheider F, Witt JT, Witzig J (2018) The SCIP Optimization Suite 6.0. ZIB-Report 18-26, Zuse Institute Berlin, Takustr. 7, 14195 Berlin, URL <http://nbn-resolving.de/urn:nbn:de:0297-zib-69361>.
- Gupta OK, Ravindran A (1985) Branch and bound experiments in convex nonlinear integer programming. *Management science* 31(12):1533–1546.
- Horst R, Tuy H (2013) *Global optimization: Deterministic approaches* (Springer Science & Business Media).
- Jiang Y, Richards T, Richards B (1994) No-good backmarking with min-conflict repair in constraint satisfaction and optimization. *PPCP*, volume 94, 2–4 (Citeseer).
- Kallrath J (2005) Solving planning and design problems in the process industry using mixed integer and global optimization. *Annals of Operations Research* 140(1):339–373, ISSN 1572-9338, URL <http://dx.doi.org/10.1007/s10479-005-3976-2>.
- Kellner K, Pfetsch ME, Theobald T (2019) Irreducible infeasible subsystems of semidefinite systems. *Journal of Optimization Theory and Applications* 181(3):727–742, URL <http://dx.doi.org/10.1007/s10957-019-01480-4>.
- Khachiyan LG (1979) A polynomial algorithm in linear programming. *Doklady Akademii Nauk SSSR*, volume 244, 1093–1096.
- Kılınc Karzan F, Nemhauser GL, Savelsbergh MWP (2009) Information-based branching schemes for binary linear mixed-integer programs. *Mathematical Programming Computation* 1(4):249–293, URL <http://dx.doi.org/http://dx.doi.org/10.1007/s12532-009-0009-1>.
- Kocis GR, Grossmann IE (1988) Global optimization of nonconvex mixed-integer nonlinear programming (MINLP) problems in process synthesis. *Industrial & engineering chemistry research* 27(8):1407–1421.
- Kronqvist J, Bernal D, Lundell A, Grossmann I (2018) A review and comparison of solvers for convex MINLP. *Preprint, Optimization Online* .

- Kuhn HW, Tucker AW (1951) Nonlinear programming. Neyman J, ed., *Proceedings of the second Berkeley symposium on mathematical statistics and probability* (University of California Press, Berkeley).
- Land AH, Doig AG (1960) An automatic method of solving discrete programming problems. *Econometrica* 28(3):497–520, URL <http://dx.doi.org/10.2307/1910129>.
- Lemke CE (1954) The dual method of solving the linear programming problem. *Naval Research Logistics Quarterly* 1(1):36–47, URL <http://dx.doi.org/10.1002/nav.3800010107>.
- Liberti L, Lavor C, Maculan N (2008) A branch-and-prune algorithm for the molecular distance geometry problem. *International Transactions in Operational Research* 15(1):1–17, URL <http://dx.doi.org/10.1111/j.1475-3995.2007.00622.x>.
- Liberti L, Lavor C, Maculan N, Nascimento MAC (2009) Reformulation in mathematical programming: An application to quantum chemistry. *Discrete Applied Mathematics* 157(6):1309 – 1318, ISSN 0166-218X, URL <http://dx.doi.org/https://doi.org/10.1016/j.dam.2007.08.044>.
- Liberti L, Pantelides CC (2003) Convex envelopes of monomials of odd degree. *Journal of Global Optimization* 25(2):157–168, URL <http://dx.doi.org/10.1023/A:1021924706467>.
- Lodi A, Tramontani A (2013) Performance variability in mixed-integer programming. *Theory Driven by Influential Applications*, 1–12 (INFORMS), URL <http://dx.doi.org/10.1287/educ.2013.0112>.
- Marques-Silva JP, Sakallah K (1999) Grasp: A search algorithm for propositional satisfiability. *Computers, IEEE Transactions on* 48(5):506–521, URL <http://dx.doi.org/10.1109/12.769433>.
- McCormick GP (1976) Computability of global solutions to factorable nonconvex programs: Part I—Convex underestimating problems. *Mathematical programming* 10(1):147–175, URL <http://dx.doi.org/10.1007/bf01580665>.
- McGill R, Tukey JW, Larsen WA (1978) Variations of box plots. *The American Statistician* 32(1):12–16, URL <http://dx.doi.org/10.2307/2683468>.
- Mehrotra S (1992) On the implementation of a primal-dual interior point method. *SIAM Journal on optimization* 2(4):575–601, URL <http://dx.doi.org/10.1137/0802028>.
- Mészáros C (1999) The BPMPD interior point solver for convex quadratic problems. *Optimization Methods and Software* 11(1-4):431–449, URL <http://dx.doi.org/10.1080/10556789908805758>.
- Misener R, Floudas CA (2014) ANTIGONE: algorithms for continuous/integer global optimization of nonlinear equations. *Journal of Global Optimization* 59(2-3):503–526, URL <http://dx.doi.org/10.1007/s10898-014-0166-2>.
- Murty KG, Yu FT (1988) *Linear complementarity, linear and nonlinear programming*, volume 3 (Heldermann).
- Nocedal J, Wright SJ (2006) *Nonlinear Equations* (Springer), ISBN 0387400656.
- Phillips AT, Rosen JB (1994) A quadratic assignment formulation of the molecular conformation problem. *Journal of Global Optimization* 4(2):229–241, ISSN 1573-2916, URL <http://dx.doi.org/10.1007/BF01096724>.

- Quesada I, Grossmann IE (1992) An LP/NLP based branch and bound algorithm for convex MINLP optimization problems. *Computers & chemical engineering* 16(10-11):937–947, URL [http://dx.doi.org/10.1016/0098-1354\(92\)80028-8](http://dx.doi.org/10.1016/0098-1354(92)80028-8).
- Sandholm T, Shields R (2006) Nogood learning for mixed integer programming. *Workshop on Hybrid Methods and Branching Rules in Combinatorial Optimization, Montréal, Concordia University, Canada*.
- Stallman RM, Sussman GJ (1977) Forward reasoning and dependency-directed backtracking in a system for computer-aided circuit analysis. *Artificial Intelligence* 9(2):135 – 196, ISSN 0004-3702, URL [http://dx.doi.org/https://doi.org/10.1016/0004-3702\(77\)90029-7](http://dx.doi.org/https://doi.org/10.1016/0004-3702(77)90029-7).
- Vavasis SA (1995) Complexity issues in global optimization: a survey. *Handbook of global optimization*, 27–41 (Springer), URL [http://dx.doi.org/10.1007/978-1-4615-2025-2\\_2](http://dx.doi.org/10.1007/978-1-4615-2025-2_2).
- Vigerske S, Gleixner A (2018) SCIP: global optimization of mixed-integer nonlinear programs in a branch-and-cut framework. *Optimization Methods and Software* 33(3):563–593, URL <http://dx.doi.org/10.1080/10556788.2017.1335312>.
- Viswanathan J, Grossmann I (1990) A combined penalty function and outer-approximation method for minlp optimization. *Computers & Chemical Engineering* 14(7):769 – 782, ISSN 0098-1354, URL [http://dx.doi.org/https://doi.org/10.1016/0098-1354\(90\)87085-4](http://dx.doi.org/https://doi.org/10.1016/0098-1354(90)87085-4).
- Wächter A, Biegler LT (2006) On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming* 106(1):25–57, URL <http://dx.doi.org/10.1007/s10107-004-0559-y>.
- Westerlund T, Pettersson F (1995) An extended cutting plane method for solving convex minlp problems. *Computers & Chemical Engineering* 19:131 – 136, ISSN 0098-1354, URL [http://dx.doi.org/https://doi.org/10.1016/0098-1354\(95\)87027-X](http://dx.doi.org/https://doi.org/10.1016/0098-1354(95)87027-X).
- Witzig J, Beckenbach I, Eifler L, Fackeldey K, Gleixner A, Grever A, Weber M (2018) Mixed-integer programming for cycle detection in nonreversible markov processes. *Multiscale Modeling & Simulation* 16(1):248–265, URL <http://dx.doi.org/10.1137/16M1091162>.
- Witzig J, Berthold T, Heinz S (2017) Experiments with conflict analysis in mixed integer programming. *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, 14th International Conference, CPAIOR 2017*, volume 10335 of *LNCS*, 211–220 (Springer Berlin Heidelberg), URL [http://dx.doi.org/10.1007/978-3-319-59776-8\\_17](http://dx.doi.org/10.1007/978-3-319-59776-8_17).
- Witzig J, Berthold T, Heinz S (2019a) Computational aspects of infeasibility analysis in mixed integer programming. Technical Report 19-54, Zuse Institute Berlin, Takustr. 7, 14195 Berlin, URL <http://nbn-resolving.de/urn:nbn:de:0297-zib-74962>.
- Witzig J, Berthold T, Heinz S (2019b) A status report on conflict analysis in mixed integer nonlinear programming. *Integration of AI and OR Techniques in Constraint Programming. CPAIOR 2019*, volume 11494, 84 – 94, URL [http://dx.doi.org/10.1007/978-3-030-19212-9\\_6](http://dx.doi.org/10.1007/978-3-030-19212-9_6).
- Witzig J, Gleixner A (2019) Conflict-driven heuristics for mixed integer programming. *INFORMS Journal on Computing* Accepted for publication.