A Constraint Sequent Calculus for First-Order Logic with Linear Integer Arithmetic

Philipp Rümmer

Department of Computer Science and Engineering
Chalmers University of Technology and Göteborg University
SE-412 96 Göteborg, Sweden
philipp@chalmers.se

Abstract. First-order logic modulo the theory of integer arithmetic is the basis for reasoning in many areas, including deductive software verification and software model checking. While satisfiability checking for ground formulae in this logic is well understood, it is still an open question how the general case of quantified formulae can be handled in an efficient and systematic way. As a possible answer, we introduce a sequent calculus that combines ideas from free-variable constraint tableaux with the Omega quantifier elimination procedure. The calculus is complete for theorems of first-order logic (without functions, but with arbitrary uninterpreted predicates), can decide Presburger arithmetic, and is complete for a substantial fragment of the combination of both.

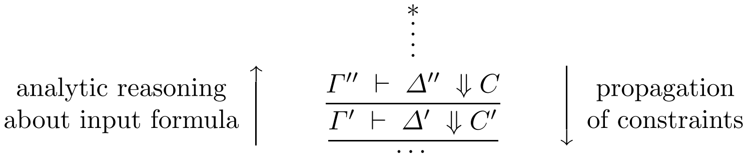
1 Introduction

One of the main challenges in automated theorem proving is to combine reasoning about full first-order logic (FOL), including quantifiers, with reasoning about theories like the integers. At the time, there are efficient provers for handling formulae in first-order logic, as well as SMT-solvers that can efficiently handle ground problems modulo many theories, but the support for the combination of both is typically weak. In this paper, we develop a novel calculus for reasoning about first-order logic modulo linear integer arithmetic that is complete for both the first-order part and the theory part, and that can handle a substantial fragment of the combination of both. Because the calculus is close to the DPLL(T) architecture, techniques and optimisations used in SMT-solvers are readily applicable when working on ground problems, but can be combined with free-variable techniques to treat quantifiers more systematically.

We start from two existing approaches: free-variable tableaux with incremental closure, following the work by Martin Giese [1], and the Omega quantifier elimination procedure [2] for deciding Presburger arithmetic (PA) [3]. From the former method, our calculus inherits the concept of generating *constraints* that describe valuations of free variables for which a formula is satisfied. The latter method provides the basic rules for dealing with linear integer arithmetic, and the concept of recursive application of a calculus in order to handle nested and alternating quantifiers. The resulting calculus accepts arbitrary formulae of PA enriched with arbitrary uninterpreted predicates as input. Uninterpreted functions are not directly supported, but can be treated by a translation to uninterpreted predicates and functionality and totality axioms.

Our calculus operates on *constrained sequents* $\Gamma \vdash \Delta \Downarrow C$, which consist of two sets Γ, Δ of formulae (the antecedent and the succedent) and one further formula C (the constraint). In this paper, C will always be a formula of PA. The semantics of a constrained sequent is the same as of the implication $C \Rightarrow (\Gamma \vdash \Delta)$, i.e., we call the sequent valid if the constraint C implies the ordinary sequent $\Gamma \vdash \Delta$ (and the ordinary sequent holds iff the formula $\bigwedge \Gamma \to \bigvee \Delta$ holds). In this sense, we can say that the constraint C is an approximation of the sequent $\Gamma \vdash \Delta$. The sequent $\forall x.(x \dot{\geq} 0 \to p(x)) \vdash p(c) \Downarrow c \dot{\geq} 0$ is valid, for instance, as are the sequents $\forall x.(x \dot{\geq} 0 \to p(x)) \vdash p(c) \Downarrow c \doteq 3$ and $\Gamma \vdash \Delta \Downarrow \mathit{false}$.

In practice, the constraints of sequents will be unknown during the construction of a proof. Reasoning about constrained sequents thus consists of two or more phases: starting with a problem $\Gamma \vdash \Delta \Downarrow ?$ with unknown constraint, a proof procedure will first apply analytic rules to the antecedent and succedent and build a proof tree, similarly as in a normal Gentzen-style sequent calculus. At some point when it seems appropriate, the procedure will start to close branches by synthesising sufficient constraints, which are subsequently propagated downwards from the leaves to the root of the tree. If the constraint that reaches the root is found to be valid, the validity of the input problem $\Gamma \vdash \Delta$ has been shown; otherwise, the procedure will continue to expand the proof tree and later update the resulting constraints.


*
analytic reasoning about input formula
$\Gamma'' \vdash \Delta'' \Downarrow C$
$\Gamma' \vdash \Delta' \Downarrow C'$
$\cdots$
propagation of constraints

In the special case that the input problem $\Gamma \vdash \Delta$ does not contain uninterpreted predicates (i.e., corresponds to a PA formula), it is always possible to find proofs such that the resulting constraint is equivalent to $\Gamma \vdash \Delta$ (we will call such proofs *exhaustive*). This makes it possible to use the calculus as a quantifier elimination procedure for PA.

Our main contributions are: the introduction of the calculus, completeness results for a number of fragments (including FOL and PA), a complete and terminating proof strategy for the PA fragment, and the result that fair proof construction is complete for formulae that are provable at all. We describe two important refinements of the calculus.

The paper is organised as follows: After giving basic definitions in Sect. 2, we introduce our calculus in three steps: Sect. 3 gives a version for pure first-order logic, Sect. 4 a minimalist version for first-order logic modulo integer arithmetic, together with completeness results, and Sect. 5 an equivalent but more refined calculus. Sect. 6 contains the result that fair proof strategies are complete. Two optimisations for the calculus are described in Sect. 7 and 8. Information about the prototypical implementation of the calculus and initial experimental results are given in Sect. 9. Finally, Sect. 10 summarises related work and Sect. 11 concludes.

2 Preliminaries

We assume that the reader is familiar with classical first-order logic and Gentzen-style sequent calculi, see [4] for an introduction. Assuming that $x \in X$ ranges over an infinite set of variables, $c \in A$ over an infinite set of constants, $p \in P$ over a set of uninterpreted predicates with fixed arity, and $\alpha \in \mathbb{Z}$ over integers, the syntactic categories of terms t and formulae ϕ are defined by:

$$\begin{aligned} t \; &::= \; \alpha \;\big|\; x \;\big|\; c \;\big|\; \alpha t + \cdots + \alpha t \\ \phi \; &::= \; \phi \wedge \phi \;\big|\; \phi \vee \phi \;\big|\; \neg\phi \;\big|\; \forall x.\phi \;\big|\; \exists x.\phi \;\big|\; t \doteq 0 \;\big|\; t \dot{\geq} 0 \;\big|\; t \dot{\leq} 0 \;\big|\; \alpha \mid t \;\big|\; p(t, \ldots, t) \end{aligned}$$

For reasons of simplicity, we only allow 0 as right-hand side of equations and inequalities, although we deviate from this convention in some places for sake of clarity. The explicit divisibility operator $\alpha \mid t$ is added for presentation purposes only and does not add any expressiveness (divisibility can also be expressed with an existentially quantified equation). Further, we use the abbreviations *true*, *false* for the equations $0 \doteq 0$, $1 \doteq 0$ and $\phi \rightarrow \psi$ as abbreviation for $\neg\phi \vee \psi$.

Simultaneous substitution of terms $t_1, \ldots, t_n$ for variables $x_1, \ldots, x_n$ is denoted by $[x_1/t_1, \ldots, x_n/t_n]\phi$, whereby we assume that variable capture is avoided by renaming bound variables when necessary. As short-hand notations, we sometimes also substitute terms for constants (as in $[c/t]\phi$), quantify over constants (as in $\forall c.\phi$), or quantify over sets of constants (as in $\forall U.\phi$).

Semantics. The only universe considered for evaluation are the integers $\mathbb{Z}$ (an exception is Sect. 3, where we treat normal first-order logic). A variable assignment $\beta : X \to \mathbb{Z}$ is a mapping from variables to integers, a constant assignment $\delta : A \to \mathbb{Z}$ a mapping from constants to integers, and an interpretation $I : P \to \mathcal{P}(\mathbb{Z}^*)$ a mapping from predicates to sets of $\mathbb{Z}$-tuples. The evaluation function $val_{I,\beta,\delta}$ for terms and formulae is then defined as is common and gives the arithmetic operations their normal meaning, for instance:

$$\begin{aligned} val_{I,\beta,\delta}(\alpha_1 t_1 + \cdots \alpha_n t_n) \; &= \; \sum_{i=1}^{n} \alpha_i \cdot val_{I,\beta,\delta}(t_i) \\ val_{I,\beta,\delta}(\alpha \mid t) \; &= \; tt \quad \text{iff there is } a \in \mathbb{Z} \text{ with } \alpha \cdot a = val_{I,\beta,\delta}(t) \\ val_{I,\beta,\delta}(p(t_1, \ldots, t_n)) \; &= \; tt \quad \text{iff } (val_{I,\beta,\delta}(t_1), \ldots, val_{I,\beta,\delta}(t_n)) \in I(p) \end{aligned}$$

We call a formula ϕ valid if $val_{I,\beta,\delta}(\phi)$ is true for all I, β, δ.

Sequents. If Γ, Δ are finite sets of formulae and C is a formula, all of which do not contain free variables, then $\Gamma \vdash \Delta$ is an (ordinary) sequent and $\Gamma \vdash \Delta \Downarrow C$ is a (constrained) sequent. We sometimes identify sequents with the formulae $\bigwedge \Gamma \to \bigvee \Delta$ (resp., $\bigwedge \Gamma \wedge C \to \bigvee \Delta$). A calculus rule is a binary relation between finite sets of constrained sequents (the premisses) and constrained sequents (the conclusion). A sequent calculus rule is called sound, iff, for all instances

$$\frac{\Gamma_1 \vdash \Delta_1 \Downarrow C_1 \quad \cdots \quad \Gamma_n \vdash \Delta_n \Downarrow C_n}{\Gamma \vdash \Delta \Downarrow C}$$

it holds that: if all premisses $\Gamma_1 \vdash \Delta_1 \Downarrow C_1, \ldots, \Gamma_n \vdash \Delta_n \Downarrow C_n$ are valid, then $\Gamma \vdash \Delta \Downarrow C$ is valid. Proof trees are defined as is common as trees growing upwards in which each node is labelled with a constrained sequent, and in which each node that is not a leaf is related with the nodes directly above through an instance of a calculus rule. A proof is closed if it is finite, and if all leaves are justified by a rule instance without premisses.

Simplification. We denote elementary simplification steps on terms and atomic formulae in a proof with SIMP, without showing more details about the applied transformation (in an implementation, SIMP might be a part of the datastructures for formulae). SIMP normalises terms to the form $\alpha_1 t_1 + \cdots + \alpha_n t_n$, in which $\alpha_1, \ldots, \alpha_n$ are non-zero integers and $t_1, \ldots, t_n$ are pairwise distinct variables, constants, or 1 (possibly 0 as the empty sum). Further, terms are put into a canonical form by sorting summands according to a well-founded ordering $<_r$:

- on variables, constants and integers, $<_r$ is an arbitrary well-ordering such that variables are bigger than constants, constants are bigger than integers, and: $0 <_r 1 <_r -1 <_r 2 <_r -2 <_r 3 <_r \cdots$.
- on terms with coefficients, $<_r$ is defined by $\alpha t <_r \alpha' t'$ if and only if $t <_r t'$ or $t = t'$ and $\alpha <_r \alpha'$.
- on linear combinations, $<_r$ is defined by $\alpha_1 t_1 + \cdots + \alpha_n t_n <_r \alpha'_1 t'_1 + \cdots + \alpha'_k t'_k$ if and only if $\{\!\{\alpha_1 t_1, \ldots, \alpha_n t_n\}\!\} <_r \{\!\{\alpha'_1 t'_1, \ldots, \alpha'_n t'_n\}\!\}$ (in the multiset extension of $<_r$, cf. [5]).

Atomic formulae $t \doteq 0$, $t \mathrel{\dot{\geq}} 0$, $t \mathrel{\dot{\leq}} 0$ are normalised by SIMP such that the coefficients of non-constant terms in t are coprime (do not have non-trivial factors in common), and such that the leading coefficient is non-negative. This also detects that equations like $2y - 6c + 1 \doteq 0$ are unsolvable and equivalent to *false*, and that an inequality like $2y - 6c + 1 \mathrel{\dot{\leq}} 0$ can be simplified and rounded to $y - 3c + 1 \mathrel{\dot{\leq}} 0$ thanks to the discreteness of the integers. All inequalities in the succedent are moved to the antecedent. A divisibility judgement $\alpha \mid t$ is normalised like an equation $\alpha x + t \doteq 0$, and it is ensured that α and the leading coefficient of t are positive.

3 A Constraint Sequent Calculus for First-Order Logic

We first introduce a very restricted calculus for pure first-order logic, in order to illustrate how the framework of constrained sequents is related to normal free-

$$\frac{\mathit{\Gamma} \vdash \phi, \mathit{\Delta} \Downarrow C \qquad \mathit{\Gamma} \vdash \psi, \mathit{\Delta} \Downarrow D}{\mathit{\Gamma} \vdash \phi \wedge \psi, \mathit{\Delta} \Downarrow C \wedge D}\ \textsc{and-right}$$

$$\frac{\mathit{\Gamma}, \phi \vdash \mathit{\Delta} \Downarrow C \qquad \mathit{\Gamma}, \psi \vdash \mathit{\Delta} \Downarrow D}{\mathit{\Gamma}, \phi \vee \psi \vdash \mathit{\Delta} \Downarrow C \wedge D}\ \textsc{or-left}$$

$$\frac{\mathit{\Gamma}, \phi, \psi \vdash \mathit{\Delta} \Downarrow C}{\mathit{\Gamma}, \phi \wedge \psi \vdash \mathit{\Delta} \Downarrow C}\ \textsc{and-left} \qquad \frac{\mathit{\Gamma} \vdash \phi, \psi, \mathit{\Delta} \Downarrow C}{\mathit{\Gamma} \vdash \phi \vee \psi, \mathit{\Delta} \Downarrow C}\ \textsc{or-right}$$

$$\frac{\mathit{\Gamma} \vdash \phi, \mathit{\Delta} \Downarrow C}{\mathit{\Gamma}, \neg\phi \vdash \mathit{\Delta} \Downarrow C}\ \textsc{not-left} \qquad \frac{\mathit{\Gamma}, \phi \vdash \mathit{\Delta} \Downarrow C}{\mathit{\Gamma} \vdash \neg\phi, \mathit{\Delta} \Downarrow C}\ \textsc{not-right}$$

$$\frac{\mathit{\Gamma} \vdash [x/c]\phi, \exists x.\phi, \mathit{\Delta} \Downarrow [x/c]C}{\mathit{\Gamma} \vdash \exists x.\phi, \mathit{\Delta} \Downarrow \exists x.C}\ \textsc{ex-right} \qquad \frac{\mathit{\Gamma}, [x/c]\phi, \forall x.\phi \vdash \mathit{\Delta} \Downarrow [x/c]C}{\mathit{\Gamma}, \forall x.\phi \vdash \mathit{\Delta} \Downarrow \exists x.C}\ \textsc{all-left}$$

$$\frac{\mathit{\Gamma} \vdash [x/c]\phi, \mathit{\Delta} \Downarrow [x/c]C}{\mathit{\Gamma} \vdash \forall x.\phi, \mathit{\Delta} \Downarrow \forall x.C}\ \textsc{all-right} \qquad \frac{\mathit{\Gamma}, [x/c]\phi \vdash \mathit{\Delta} \Downarrow [x/c]C}{\mathit{\Gamma}, \exists x.\phi \vdash \mathit{\Delta} \Downarrow \forall x.C}\ \textsc{ex-left}$$

Fig. 1. The rules for first-order predicate logic (without equality). In all rules, c is a constant that does not occur in the conclusion: in contrast to the usage of Skolem functions and free variables in tableaux, the same kinds of symbols (constants) are used to handle both existential and universal quantifiers. Arbitrary renaming of bound variables is allowed in the constraints when necessary to avoid variable capture.

variable tableau calculi. This section is exceptional in that we do *not* assume evaluation of formulae over the universe $\mathbb{Z}$ of integers, and that we allow equations $s \doteq t$ whose right-hand side is not 0. The rules from Fig. 1, together with the following closure rule, form the calculus Pred^C:

$$\frac{*}{\mathit{\Gamma}, p(s_1, \ldots, s_n) \vdash p(t_1, \ldots, t_n), \mathit{\Delta} \Downarrow \bigwedge_i s_i \doteq t_i}\ \textsc{pred-close}$$

Instead of unifying complementary literals, a conjunction of equations about the predicate arguments is generated and propagated as a constraint.

Example 1. We show a proof for the sequent $\forall x.\exists y.p(x,y) \vdash \exists z.p(a,z)$. In order to instantiate existential and universal quantifiers, fresh constants c, d, e are introduced. The constraints on the right-hand side are practically filled in *after* applying PRED-CLOSE. Because $\exists x.\forall y.\exists z.(x \doteq a \wedge y \doteq z)$ is valid, also the validity of the original problem is proven.

$$\dfrac{\dfrac{\dfrac{\dfrac{*}{\ldots, p(c,d) \vdash \ldots, p(a,e) \Downarrow c \doteq a \wedge d \doteq e}\ \textsc{pred-close}}{\ldots, p(c,d) \vdash \exists z.p(a,z) \Downarrow \exists z.(c \doteq a \wedge d \doteq z)}\ \textsc{ex-right}}{\ldots, \exists y.p(c,y) \vdash \exists z.p(a,z) \Downarrow \forall y.\exists z.(c \doteq a \wedge y \doteq z)}\ \textsc{ex-left}}{\forall x.\exists y.p(x,y) \vdash \exists z.p(a,z) \Downarrow \exists x.\forall y.\exists z.(x \doteq a \wedge y \doteq z)}\ \textsc{all-left}$$

It is easy to see that a constraint C produced by a proof can only consist of equations over variables and constants, conjunctions, and quantifiers (because these are the only constructs that are introduced in constraints by the rules of Pred^C). The validity of constraints/formulae of this kind is decidable and corresponds to simultaneous unification, which makes the calculus effective.

Proof on page 190

Lemma 2 (Soundness). *If a sequent $\Gamma \vdash \Delta \Downarrow C$ is provable in Pred^C, then it is valid (holds in all first-order structures).*

Proof on page 190

Lemma 3 (Completeness). *Suppose ϕ is closed, valid (holds in all first-order structures), and does not contain constants. Then there is a valid constraint C such that $\vdash \phi \Downarrow C$ is provable in Pred^C.*

It can be observed that Pred^C is also proof confluent, which strengthens Lem. 3. In order to continue ("complement") a partial proof, it can be both necessary to expand branches further and to update constraints anywhere in the proof:

Proof on page 195

Lemma 4 (Proof confluence). *Suppose that ϕ is valid. Any (partial) Pred^C-proof with root $\vdash \phi \Downarrow ?$ that does not contain applications of* PRED-CLOSE *can be complemented to a closed proof tree with root $\vdash \phi \Downarrow C$ for some valid constraint C.*

4 Adding Integer Arithmetic

Relatively few changes to the calculus Pred^C from the previous section are necessary to reason about problems in integer arithmetic. In this section, we describe a minimalist approach in which all integer reasoning happens during the constraint solving and investigate fragments on which the resulting method is complete. Later in the paper, the calculus is refined and optimised. From now on and in contrast to the previous section, assume that formulae and terms are evaluated over first-order structures with the universe $\mathbb{Z}$ as described in Sect. 2.

In contrast to the previous section, to handle integer arithmetic disjunctive constraints also need to be considered. We thus split the rule PRED-CLOSE into two new rules, one of which (PRED-UNIFY) generates unification conditions for complementary pairs, while the other one (CLOSE) allows to synthesise a constraint from arbitrary formulae in a sequent:

$$\frac{\Gamma, p(s_1, \ldots, s_n) \vdash p(t_1, \ldots, t_n), \bigwedge_i s_i - t_i \doteq 0, \Delta \Downarrow C}{\Gamma, p(s_1, \ldots, s_n) \vdash p(t_1, \ldots, t_n), \Delta \Downarrow C} \ \textsc{pred-unify}$$

$$\frac{*}{\Gamma, \phi_1, \ldots, \phi_n \vdash \psi_1, \ldots, \psi_m, \Delta \Downarrow \neg\phi_1 \vee \cdots \vee \neg\phi_n \vee \psi_1 \vee \cdots \vee \psi_m} \ \textsc{close}$$

($\phi_1, \ldots, \phi_n, \psi_1, \ldots, \psi_m$ do not contain uninterpreted predicates)

Besides these two rules, PresPred^C_S contains all rules given in Fig. 1. It is obvious that any proof in Pred^C can be translated to a proof in PresPred^C_S by replacing applications of PRED-CLOSE with applications of PRED-UNIFY, followed by CLOSE, which means that PresPred^C_S is complete for first-order logic.

Because uninterpreted predicates are excluded in CLOSE, the constraint resulting from a proof is always a formula in Presburger arithmetic and can in principle be handled using any decision procedure for PA (e.g., [6, 2]). We come back to this issue later in the paper and assume for the time being that some procedure is available for deciding the validity of constraints.

As an implication of a more general result (Lem. 17), it can be observed that $\mathrm{PresPred}_S^C$ is proof-confluent: if ϕ is provable, then every partial proof of $\vdash \phi \Downarrow ?$ can be extended to a closed proof of a sequent $\vdash \phi \Downarrow C$ with valid constraint C.

Example 5. We show a proof for the following sequent:

$$\forall x.p(2x), \forall x.\neg p(2x+1) \vdash \forall y.(p(y) \to p(y+10))$$

This is done by first building the "main proof" (upwards) to a point where CLOSE can be applied. The constraints $C_1, \ldots, C_4$ are then filled in and propagated downwards:

$$
\begin{array}{cl}
* & \\ \cline{1-1}
\ldots \vdash \ldots, 2d - c - 10 \doteq 0, c - 2e - 1 \doteq 0 \Downarrow C_1 & \textsc{close} \\ \cline{1-1}
p(2d), \ldots, p(c) \vdash p(c+10), p(2e+1) \Downarrow C_1 & \textsc{pred-unify} \times 2 \\ \cline{1-1}
\ldots, p(2d), \forall x.\neg p(2x+1), p(c) \vdash p(c+10) \Downarrow C_2 & \textsc{all-left}, \textsc{not-left} \\ \cline{1-1}
\forall x.p(2x), \forall x.\neg p(2x+1), p(c) \vdash p(c+10) \Downarrow C_3 & \textsc{all-left} \\ \cline{1-1}
\forall x.p(2x), \forall x.\neg p(2x+1) \vdash \neg p(c) \vee p(c+10) \Downarrow C_3 & \textsc{or-right}, \textsc{not-right} \\ \cline{1-1}
\forall x.p(2x), \forall x.\neg p(2x+1) \vdash \forall y.(p(y) \to p(y+10)) \Downarrow C_4 & \textsc{all-right}
\end{array}
$$

The constraints are:

$$
\begin{array}{lcllll}
C_1 & = & & & 2d - c - 10 \doteq 0 \vee c - 2e - 1 \doteq 0 \\
C_2 & = & \exists y.[e/y]C_1 & = & \exists y.(2d - c - 10 \doteq 0 \vee c - 2y - 1 \doteq 0) \\
C_3 & = & \exists x.[d/x]C_2 & = & \exists x.\exists y.(2x - c - 10 \doteq 0 \vee c - 2y - 1 \doteq 0) \\
 & & & \equiv & 2 \mid (c+10) \vee 2 \mid (c-1) \\
C_4 & = & \forall x.[c/x]C_3 & = & \forall x.(2 \mid (x+10) \vee 2 \mid (x-1)) \\
 & & & \equiv & \mathit{true}
\end{array}
$$

Because C_4 is valid, we have proven the validity of the original formula. The simplification of constraints is explained in more detail later.

Completeness on fragments. Two fragments on which $\mathrm{PresPred}_S^C$ is complete are the classes of purely universal and of purely existential formulae. We call positions in the antecedent/succedent of a sequent *positive* if they are underneath an odd/even number of negations. All other positions are called *negative.*

Lemma 6. *If $\Gamma \vdash \Delta$ is a valid sequent in which $\exists$ only occurs in negative and $\forall$ only in positive positions, then there is a valid PA constraint C such that $\Gamma \vdash \Delta \Downarrow C$ has a proof in the calculus $\mathit{PresPred}_S^C$.* Proof on page 195

Lemma 7. *If $\Gamma \vdash \Delta$ is a valid sequent (without constants) in which $\exists$ only occurs in positive and $\forall$ only in negative positions, then there is a valid PA constraint C such that $\Gamma \vdash \Delta \Downarrow C$ has a proof in the calculus $\mathit{PresPred}_S^C$.* Proof on page 196

Comparison with $\mathcal{ME}$(LIA). We can also show that the calculus PresPred^C_S is complete on the fragment of first-order logic modulo linear integer arithmetic that can be handled by Model Evolution modulo linear integer arithmetic [7]. Ignoring minor syntactic issues and the fact that $\mathcal{ME}$(LIA) works on clauses, $\mathcal{ME}$(LIA) is a sound and complete calculus for proving the unsatisfiability of formulae of the shape $\exists\bar{a}.(\phi \wedge \psi)$, where:

- $\bar{a} = (a_1, \ldots, a_m)$ is a vector of existentially quantified variables,
- ϕ is a formula of Presburger arithmetic over $\bar{a}$ that only has finitely many solutions, and
- ψ is an arbitrary formula over $\bar{a}$ in which $\exists$ only occurs in negative and $\forall$ only in positive positions.

Proof on page 197

Lemma 8. *If $\exists\bar{a}.(\phi \wedge \psi)$ as above is an unsatisfiable formula that does not contain constants or free variables, then there is a valid constraint C such that the sequent $\exists\bar{a}.(\phi \wedge \psi) \vdash \;\Downarrow C$ has a proof in PresPred^C_S.*

5 Built-In Handling of Presburger Arithmetic

Although the calculus from the previous section is in principle usable, it practically has a number of shortcomings: the handling of arithmetic in constraints provides little guidance for the construction of proofs, so that large constraints are produced in a very indeterministic manner that cannot be solved efficiently. Moreover, constraints are even needed to handle ground problems, for which branch-local reasoning should be sufficient. The main goal when refining the calculus is, therefore, to reduce the usage of constraints as far as possible.

In this section, we define built-in rules for handling linear integer arithmetic that can be interleaved with the rules from the previous section. The rules make it possible to handle ground problems branch-locally: proof trees for ground problems can be constructed depth-first (non-iteratively). Together with the refinement in Sect. 7, it can be achieved that the only constraints that can result from a subproof in case of ground problems are *true* or *false*. More generally, branch-local reasoning is possible for innermost $\forall$-quantifiers in positive and $\exists$ in negative positions. The arithmetic rules also yield a decision procedure for Presburger arithmetic that can be used to decide constraints (Sect. 5.3).

The rules in detail. The calculus PresPred^C consists of the rules given in Fig. 2, together with all rules from the calculus PresPred^C_S and the simplification rule SIMP. We introduce new rules EX-RIGHT-D, ALL-LEFT-D that instantiate quantified formulae destructively, because formulae that do not contain uninterpreted predicates never have to be instantiated twice (also see Lem. 17 below).

The equality handling follows the calculus given in [8] and can solve arbitrary equations in the antecedent, in the sense that the equations are rewritten until the leading coefficients are all 1 and the leading terms of equations occur in exactly one place. Speaking in terms of matrices, RED is the rule for performing

$$\frac{\varGamma \ \vdash \ [x/c]\phi, \varDelta \ \Downarrow [x/c]C}{\varGamma \ \vdash \ \exists x.\phi, \varDelta \ \Downarrow \exists x.C}\ \textsc{ex-right-d} \qquad \frac{\varGamma, [x/c]\phi \ \vdash \ \varDelta \ \Downarrow [x/c]C}{\varGamma, \forall x.\phi \ \vdash \ \varDelta \ \Downarrow \exists x.C}\ \textsc{all-left-d}$$

(c a constant that does not occur in the conclusion,
ϕ does not contain uninterpreted predicates)

$$\frac{\varGamma, t \doteq 0 \ \vdash \ \phi[s + \alpha \cdot t], \varDelta \ \Downarrow C}{\varGamma, t \doteq 0 \ \vdash \ \phi[s], \varDelta \ \Downarrow C}\ \textsc{red}$$

$$\frac{\varGamma, \alpha(u + c') + t \doteq 0, c - u - c' \doteq 0 \ \vdash \ \varDelta \ \Downarrow [x/c']C}{\varGamma, \alpha c + t \doteq 0 \ \vdash \ \varDelta \ \Downarrow \forall x.C}\ \textsc{col-red}$$

(c' a constant that does not occur in the conclusion or in u)

$$\frac{\varGamma, \alpha(u + c') + t \doteq 0, c - u - c' \doteq 0 \ \vdash \ \varDelta \ \Downarrow [x/c']C}{\varGamma, \alpha c + t \doteq 0 \ \vdash \ \varDelta \ \Downarrow [x/c - u]C}\ \textsc{col-red-subst}$$

(c' a constant that does not occur in the conclusion or in u)

$$\frac{\varGamma, \exists x.\alpha x + t \doteq 0 \ \vdash \ \varDelta \ \Downarrow C}{\varGamma, \alpha \mid t \ \vdash \ \varDelta \ \Downarrow C}\ \textsc{div-left}$$

(x an arbitrary variable)

$$\frac{\varGamma, (\alpha \mid t + 1) \vee \cdots \vee (\alpha \mid t + \alpha - 1) \ \vdash \ \varDelta \ \Downarrow C}{\varGamma \ \vdash \ \alpha \mid t, \varDelta \ \Downarrow C}\ \textsc{div-right} \qquad (\alpha > 0)$$

$$\frac{\varGamma, \alpha c - t \doteq 0 \ \vdash \ \varDelta \ \Downarrow C}{\varGamma, \alpha c - t \doteq 0 \ \vdash \ \varDelta \ \Downarrow [x/t]C' \vee \alpha \nmid t}\ \textsc{div-close}$$

(c does not occur in t or in C', C' a PA formula such that $C \Leftrightarrow [x/\alpha c]C'$)

$$\frac{\varGamma \ \vdash \ t \mathrel{\dot{\leq}} 0, \varDelta \ \Downarrow C \quad \varGamma \ \vdash \ t \mathrel{\dot{\geq}} 0, \varDelta \ \Downarrow D}{\varGamma \ \vdash \ t \doteq 0, \varDelta \ \Downarrow C \wedge D}\ \textsc{split-eq}$$

$$\frac{\varGamma, t \doteq 0 \ \vdash \ \varDelta \ \Downarrow C}{\varGamma, t \mathrel{\dot{\leq}} 0, t \mathrel{\dot{\geq}} 0 \ \vdash \ \varDelta \ \Downarrow C}\ \textsc{anti-symm}$$

$$\frac{\varGamma, \alpha c + s \mathrel{\dot{\geq}} 0, \beta c + t \mathrel{\dot{\leq}} 0, \beta s - \alpha t \mathrel{\dot{\geq}} 0 \ \vdash \ \varDelta \ \Downarrow C}{\varGamma, \alpha c + s \mathrel{\dot{\geq}} 0, \beta c + t \mathrel{\dot{\leq}} 0 \ \vdash \ \varDelta \ \Downarrow C}\ \textsc{fm-elim}$$

$$(\alpha > 0,\ \beta > 0)$$

$$\frac{\varGamma, \begin{array}{c} \bigwedge_{i,j} \alpha_i b_j - a_i \beta_j - (\alpha_i - 1)(\beta_j - 1) \mathrel{\dot{\geq}} 0 \\ \vee \\ \bigvee_i \bigvee_{k=0}^{m_i} \left(\begin{array}{c} \alpha_i c - a_i - k \doteq 0 \ \wedge \\ \bigwedge_i \alpha_i c - a_i \mathrel{\dot{\geq}} 0 \wedge \bigwedge_j \beta_j c - b_j \mathrel{\dot{\leq}} 0 \end{array} \right) \end{array} \ \vdash \ \varDelta \ \Downarrow C}{\varGamma, \{\alpha_i c - a_i \mathrel{\dot{\geq}} 0\}_i, \{\beta_j c - b_j \mathrel{\dot{\leq}} 0\}_j \ \vdash \ \varDelta \ \Downarrow C}\ \textsc{omega-elim}$$

$$(\alpha_i > 0,\ \beta_j > 0)$$

Fig. 2. The rules for linear integer equations, inequalities, and divisibility judgements. In the rule RED, we write $\phi[s]$ in the succedent to denote that the term s occurs in an arbitrary formula in the sequent, which can in particular also be in the antecedent. m_i in OMEGA-ELIM as on page 176.

row operations, while COL-RED(-SUBST) is responsible for column operations. We define a suitable strategy for guiding the rules below.

The rules DIV-RIGHT and DIV-LEFT translate divisibility statements to equations, while DIV-CLOSE synthesises divisibility statements from equations. The formula C' in DIV-CLOSE can be found through pseudo-division (multiplying equations, inequalities or divisibility statements in C with non-zero factors). For $C = (c + d \doteq 0)$ and $\alpha = 3$, for instance, we would choose $C' = (x + 3d \doteq 0)$.

Inequalities are handled based on the Omega test [2], which is an extension of the Fourier-Motzkin variable elimination method (cf. [9]) for integer problems. The central rule is OMEGA-ELIM for replacing a conjunction of inequalities with a disjunction over simpler cases. The literal m_i in the rule is defined by:

$$m = \max_j \beta_j, \qquad m_i = \left\lfloor \frac{m\alpha_i - \alpha_i - m}{m} \right\rfloor$$

In case there are no upper bounds, we define $m = m_i = -1$. OMEGA-ELIM is directly based on the main theorem [2] underlying the Omega test, which is the following (we use the notation from [10] where also a proof is provided).

Theorem 9 (Pugh, 1992). *Suppose $L(x) = \bigwedge_i a_i \leq \alpha_i x$ is a conjunction of lower bounds and $U(x) = \bigwedge_j \beta_j x \leq b_i$ is a conjunction of upper bounds, in which all α_i and β_j are positive integers and a_i, b_j are arbitrary terms that do not contain x. Then:*

$$\exists x. L(x) \land U(x) \quad \Longleftrightarrow \quad \begin{array}{c} \bigwedge_{i,j} (\alpha_i - 1)(\beta_j - 1) \leq \alpha_i b_j - a_i \beta_j \\ \lor \\ \bigvee_i \bigvee_{k=0}^{m_i} \exists x. \big(\alpha_i x = a_i + k \land L(x) \land U(x)\big) \end{array}$$

Appealing to a geometric interpretation, the first disjunct on the right-hand side is called the "dark shadow," whereas the existentially quantified disjuncts are called "splinters." In case all of the α_is or all of the β_js are 1, the equivalence boils down to the normal Fourier-Motzkin rule:

$$\exists x. L(x) \land U(x) \quad \Longleftrightarrow \quad \bigwedge_{i,j} a_i \beta_j \leq \alpha_i b_j$$

The application of OMEGA-ELIM is only meaningful if c does not occur in formulae other than inequalities. Note, that if there are no lower or no upper bounds, the rule will replace inequalities whose leading term is c with *true*.

Because we avoid the application of OMEGA-ELIM in certain common situations (for instance, whenever the constant c occurs as argument of uninterpreted predicates), we also introduce a rule FM-ELIM for normal Fourier-Motzkin elimination. FM-ELIM can be applied with higher priority than OMEGA-ELIM and is often able to close proofs faster than OMEGA-ELIM, reducing the need to resort to the more complex rule. Further, we define two rules to convert between equations and inequalities. While the rule SPLIT-EQ is strictly necessary for certain problems, ANTI-SYMM is introduced only for reasons of efficiency.

Proof on page 201

Lemma 10 (Soundness). *If a sequent $\Gamma \vdash \Delta \Downarrow C$ is provable in $PresPred^C$, then it is valid.*

5.1 Exhaustive Proofs

The existence of a closed proof for a sequent $\Gamma \vdash \Delta \Downarrow C$ guarantees that the implication $C \Rightarrow (\Gamma \vdash \Delta)$ holds (this is the soundness of the calculus, Lem. 10). In the special case that the sequent $\Gamma \vdash \Delta$ does not contain uninterpreted predicates, it is possible to distinguish particular closed proofs that also guarantee the opposite implication $(\Gamma \vdash \Delta) \Rightarrow C$, and thus $(\Gamma \vdash \Delta) \Leftrightarrow C$. While this can be achieved in a trivial way by always applying CLOSE such that *all* formulae in a sequent are selected, it is sufficient to impose a weaker condition on proof trees that leads to smaller constraints and also makes it possible to eliminate quantifiers (Sect. 5.3). To this end, it is necessary to remember whether a constant was introduced by an existential rule (like EX-RIGHT) or a universal rule (like ALL-RIGHT), and whether other existential rules were applied in between.

Assume that a PresPred^C-proof is given. We annotate the sequents in the proof with sets U of "universal" constants that the calculus attempts to eliminate. More formally, the proof is called *exhaustive* iff there is a mapping from proof nodes (constrained sequents) to sets U of constants subject to the following conditions:

1. The rules AND-*, OR-*, NOT-*, PRED-UNIFY, RED, DIV-*, SPLIT-EQ, ANTI-SYMM, FM-ELIM, and SIMP keep or reduce the set: if the conclusion is annotated with U, the premisses are annotated with arbitrary subsets of U.
2. The rules EX-RIGHT(-D), ALL-LEFT(-D) erase the set: the premiss is annotated with $\emptyset$.
3. The rules EX-LEFT and ALL-RIGHT may add the introduced constant c to the set: if the conclusion is annotated with U, then the premiss is annotated with a subset of $U \cup \{c\}$.
4. The rule COL-RED is only applied if the conclusion is annotated with U such that $c \in U$. In this case, the premiss is annotated with a subset of $U \cup \{c'\}$.
5. The rule COL-RED-SUBST is only applied if the conclusion is annotated with U such that $c \notin U$, and if u does not contain any constants from U. In this case, the premiss is annotated with a subset of U.
6. The rule OMEGA-ELIM is only applied if the conclusion is annotated with U such that $c \in U$ and if c does not occur in Γ or Δ. In this case, the premisses are annotated with an arbitrary subset of U.
7. The rule DIV-CLOSE is only applied if the conclusion is annotated with U such that $c \in U$. In this case, the premiss is annotated with a subset of U.
8. The rule CLOSE is always applied such that all formulae without uninterpreted predicates are selected, apart from (possibly) those equations in the succedent that contain constants from U that exclusively occur in equations in the succedent.

Lemma 11 (Constraint completeness). *Suppose that a PresPred^C-proof is closed and exhaustive. For each sequent $\Gamma \vdash \Delta \Downarrow C$ in the tree that is annotated with a set U, let Γ_p, Δ_p denote the sets of PA formulae contained in Γ, Δ. The following implication holds for each sequent:*

Proof on page 202

$$\forall U.\ (\Gamma_p \vdash \Delta_p) \ \Rightarrow\ \forall U.\ C \tag{1}$$

Example 12. The formula $\neg\exists x.\exists y.(2x - c - 10 \doteq 0 \vee 2y - c + 1 \doteq 0)$ from Example 5 is simplified by constructing a proof. To see that the proof is exhaustive, the sequent with constraint D_5 is annotated with $\emptyset$, the sequent with D_1 with $\{e\}$, the sequent with D_3 with $\{d\}$, and all other sequents with the set $\{d, e\}$.

$$\dfrac{\dfrac{\dfrac{\dfrac{*}{c - 2d + 10 \doteq 0 \ \vdash \ \ \Downarrow D_1}\ \textsc{close}}{2d - c - 10 \doteq 0 \ \vdash \ \ \Downarrow D_2}\ \textsc{div-close} \qquad \dfrac{\dfrac{*}{c - 2e - 1 \doteq 0 \ \vdash \ \ \Downarrow D_3}\ \textsc{close}}{2e - c + 1 \doteq 0 \ \vdash \ \ \Downarrow D_4}\ \textsc{div-close}}{2d - c - 10 \doteq 0 \vee 2e - c + 1 \doteq 0 \ \vdash \ \ \Downarrow D_2 \wedge D_4}\ \textsc{or-left}}{\exists x.\exists y.(2x - c - 10 \doteq 0 \vee 2y - c + 1 \doteq 0) \ \vdash \ \ \Downarrow D_5}\ \textsc{ex-left} \times 2$$

The constraints resulting from the proof are:

$$\begin{array}{lclcl}
D_1 & = & & & c - 2d + 10 \not\doteq 0 \\
D_2 & = & [2d/c + 10]D_1 \vee 2 \nmid (c + 10) & = & c - (c + 10) + 10 \not\doteq 0 \vee 2 \nmid (c + 10) \\
 & & & \equiv & 2 \nmid (c + 10) \\
D_3 & = & & & c - 2e - 1 \not\doteq 0 \\
D_4 & = & [2e/c - 1]D_3 \vee 2 \nmid (c - 1) & = & c - (c - 1) - 1 \not\doteq 0 \vee 2 \nmid (c - 1) \\
 & & & \equiv & 2 \nmid (c - 1) \\
D_5 & = & \exists x.[d/x]\exists y.[e/y](D_2 \wedge D_4) & = & \exists x.\exists y.(2 \nmid (c + 10) \wedge 2 \nmid (c - 1)) \\
 & & & \equiv & 2 \nmid (c + 10) \wedge 2 \nmid (c - 1)
\end{array}$$

Because the proof is exhaustive, we know that the original formula is equivalent to $2 \nmid (c + 10) \wedge 2 \nmid (c - 1)$.

Example 13. The constraint $\forall x.(2 \mid (x + 10) \vee 2 \mid (x - 1))$ from Example 5 is simplified to *true* by constructing the following proof:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{*}{c + 2d' + 1 \doteq 0, d - d' + 5 \doteq 0, \mathit{false} \ \vdash \ \ \Downarrow \mathit{true}}\ \textsc{close}}{c + 2d' + 1 \doteq 0, d - d' + 5 \doteq 0, \exists y.\mathit{false} \ \vdash \ \ \Downarrow \mathit{true}}\ \textsc{ex-left}}{c + 2d' + 1 \doteq 0, d - d' + 5 \doteq 0, \exists y.2y - 2d' - 1 \doteq 0 \ \vdash \ \ \Downarrow \mathit{true}}\ \textsc{simp}}{c + 2d' + 1 \doteq 0, d - d' + 5 \doteq 0, \exists y.2y + c \doteq 0 \ \vdash \ \ \Downarrow \mathit{true}}\ \textsc{red}}{2(-5 + d') + c + 11 \doteq 0, d - (-5) - d' \doteq 0, \exists y.2y + c \doteq 0 \ \vdash \ \ \Downarrow \mathit{true}}\ \textsc{simp} \times 2}{2d + c + 11 \doteq 0, \exists y.2y + c \doteq 0 \ \vdash \ \ \Downarrow \mathit{true}}\ \textsc{col-red}}{\exists x.2x + c + 11 \doteq 0, \exists y.2y + c \doteq 0 \ \vdash \ \ \Downarrow \mathit{true}}\ \textsc{ex-left}}{2 \mid (c + 11), 2 \mid c \ \vdash \ \ \Downarrow \mathit{true}}\ \textsc{div-left} \times 2}{\vdash \ 2 \mid (c + 10), 2 \mid (c - 1) \ \Downarrow \mathit{true}}\ \textsc{div-right} \times 2, \textsc{simp}$$

$$\dfrac{\vdash \ 2 \mid (c + 10), 2 \mid (c - 1) \ \Downarrow \mathit{true}}{\vdash \ \forall x.(2 \mid (x + 10) \vee 2 \mid (x - 1)) \ \Downarrow \mathit{true}}\ \textsc{all-right}, \textsc{or-right}$$

5.2 The Construction of Exhaustive Proofs for PA Problems

We define a strategy to apply the PresPred^C-rules to a sequent $\Gamma \ \vdash \ \Delta \ \Downarrow ?$ of PA formulae with unknown constraint. The strategy is guaranteed to terminate and to produce a closed and exhaustive proof, and it is deterministic in the sense that no search is required, every ordering of rule applications (that is consistent with given priorities) leads to an exhaustive proof. In order to guide the proof construction, the strategy maintains a set U of constants (which is initially empty) and a term ordering $<_r$ (as in Sect. 2) that are updated when new constants are introduced or existing constants need to be reordered. The ordering $<_r$ is always chosen such that the constants in U are bigger than all constants that are not in U. Both U and $<_r$ are branch-local: different branches in a proof tree can be built using different Us and $<_r$s.

We define the strategy by listing the rules that it applies to a proof goal with descending priority. This means that step 2 will only be carried out if step 1 is impossible, etc.

1. apply SIMP (if possible).
2. apply RED if an α exists such that $s + \alpha \cdot t <_r s$
   (and if $s \neq t$ or $\phi[s]$ is not an equation in the antecedent).
3. if the antecedent contains an equation $\alpha c + t \doteq 0$ with $\alpha > 1$, then:
   - if $c \notin U$, apply COL-RED-SUBST. The fresh constant c' is inserted in the term ordering $<_r$ such that it becomes minimal, and u is chosen such that $(\alpha u + t) = \min_{<_r} \{\alpha u' + t \mid u' \text{ a term}\}$.
   - if $c \in U$ and t contains at least one further constant from U whose coefficient is not a multiple of α, apply COL-RED. The fresh constant c' is added to U and is inserted in the term ordering $<_r$ such that it becomes smaller than all other constants in U, but bigger than all constants not in U. u is again chosen such that $(\alpha u + t) = \min_{<_r} \{\alpha u' + t \mid u' \text{ a term}\}$.
4. if the antecedent contains an equation $\alpha c + t \doteq 0$ with $c \in U$, apply DIV-CLOSE, remove c from U, and update $<_r$ such that c becomes minimal.
   (This is also possible for $\alpha = 1$)
5. if possible, apply any of the following rules:
   - ANTI-SYMM.
   - FM-ELIM, unless the result is subsumed by an existing inequality in the antecedent.
   - any of the rules AND-*, OR-*, NOT-*.
6. if possible, apply any of the following rules:
   - SPLIT-EQ: if an equation exists in the succedent that contains a constant $c \in U$, and c occurs as leading term of an inequality in the antecedent.
   - OMEGA-ELIM: if inequalities $\{\alpha_i c - a_i \dot{\geq} 0\}_i$, $\{\beta_j c - b_j \dot{\leq} 0\}_j$ occur in the antecedent and $c \in U$, and if c does not occur in any other formula.
   - ALL-RIGHT, EX-LEFT: add the fresh constant c to U and insert it into $<_r$ such that it becomes maximal.
   - EX-RIGHT-D, ALL-LEFT-D: set U to the empty set $\emptyset$ and insert c arbitrarily into $<_r$.
   - DIV-LEFT, DIV-RIGHT.
7. apply CLOSE and select exactly those formulae that do not contain constants from U or uninterpreted predicates.

The steps 1–4 of the strategy work by eliminating all U-constants that occur in equations in the antecedent. Similarly as in [8], in the antecedent only equations will be left whose leading coefficient is 1 and whose leading term does not occur in other places in the sequent anymore. The steps 5–6 handle inequalities by first applying the Fourier-Motzkin rule exhaustively, and by eliminating constants using the Omega rule whenever possible. Also quantifiers, propositional connectives and divisibility judgements are treated in step 5–6. A proof that is constructed using this procedure is shown in Example 12.

Proof on page 203

Lemma 14 (Termination and exhaustiveness). *If a sequent $\Gamma \vdash \Delta \Downarrow ?$ does not contain uninterpreted predicates, the strategy from above terminates and produces a closed exhaustive proof.*

5.3 Deciding Presburger Arithmetic by Recursive Proving

The anticipated way to decide constraints in proofs is to eliminate quantifiers already during the constraint propagation, i.e., at the points where the rules EX-RIGHT(-D), ALL-LEFT(-D), ALL-RIGHT, EX-LEFT or COL-RED are applied. When building a proof incrementally, early elimination enables a prover to identify those parts of a proof that still have an unsatisfiable constraint and thus need to be expanded further. Besides, when using the procedure from the previous section together with early elimination, it is clear that only ground constraints occur in proofs, which implies that the approach decides PA.

The calculus PresPred^C itself can be used to eliminate quantifiers. This is possible because we can observe that the strategy from the previous section is always able to eliminate one level of universal quantifiers:

Proof on page 207

Lemma 15 (Quantifier elimination). *Suppose a formula ϕ does not contain uninterpreted predicates and only universal quantifiers ($\forall$ in positive positions, $\exists$ in negative positions). The strategy from the previous section produces a proof with root $\vdash \phi \Downarrow C$ in which C does not contain quantifiers (more precisely, if C contains a quantified subformula $Qx.\psi$, then x does not occur in ψ).*

This means that, in order to eliminate universal quantifiers from a formula ϕ, we can construct an exhaustive proof with root $\vdash \phi \Downarrow C$ and extract the constraint C. Similarly, existential quantifiers can be eliminated by constructing a proof for $\phi \vdash \ \Downarrow C$ (also see Example 12).

6 Fair Construction of Proofs

We now compare the calculus PresPred^C with the more restricted calculus PresPred^C_S from Sect. 4. Because the former calculus is a superset of the latter, it is a trivial observation that any sequent provable in PresPred^C_S is also provable in PresPred^C. It is also possible to show that PresPred^C cannot prove more sequents than PresPred^C_S, which means that the two calculi are equivalent.

Proof on page 208

Lemma 16. *Suppose that a* $\mathit{PresPred}^C$*-proof for the sequent* $\Gamma \vdash \Delta \Downarrow C$ *exists. For some constraint* D *with* $C \Rightarrow D$*, there is a* $\mathit{PresPred}^C_S$*-proof of the sequent* $\Gamma \vdash \Delta \Downarrow D$*.*

Proofs in $\mathrm{PresPred}^C$ can be found by a backtracking-free fair application strategy. Rules that are specific to integer arithmetic (Fig. 2) are mostly irrelevant for this result: such rules do not hinder the construction of proofs, but their application is not necessary either. Practically, the rules can help to find shorter proofs and reduce the size of constraints involved, however.

To define the notion "fair" formally, it has to be observed that formulae in a $\mathrm{PresPred}^C$-proof can be rewritten by applying RED or SIMP. When this happens, it is possible to identify a unique successor of the modified formula in the premiss of the rule application (vice versa, a formula can have multiple predecessors because distinct formulae could become equal when applying a rule).

A *fair* $\mathit{PresPred}^C$*-proof* is a possibly infinite $\mathrm{PresPred}^C$-proof for a sequent $\Gamma \vdash \Delta \Downarrow ?$ in which all constraints are ?, such that on all branches the following conditions hold:

- *Fair treatment of formulae with uninterpreted predicates:* whenever at some point on the branch one of the rules in Fig. 1 is applicable to a formula that contains uninterpreted predicates, the rule is applied to the formula or to a successor of the formula at some later point on the branch. (This implies that ALL-LEFT and EX-RIGHT are applied infinitely often to each universally quantified formula with uninterpreted predicates).
- *Fair unification of complementary literals:* if there is a sequent on the branch of the shape $\Gamma, p(\bar{t}) \vdash p(\bar{s}), \Delta \Downarrow ?$, the rule PRED-UNIFY is applied at least once on the branch to the pair $p(\bar{t})$, $p(\bar{s})$ or to successors of these formulae.
- *Exhaustiveness:* all nodes of the proof can be annotated with sets U as described in Sect. 5.1.

We say that a constraint C is *generated* by a fair proof of $\Gamma \vdash \Delta \Downarrow ?$ if a (finite) proof for $\Gamma \vdash \Delta \Downarrow C$ can be obtained by chopping off all branches of the fair proof at some point, applying CLOSE in some way to the leaves and propagating the resulting constraints through the proof tree.

Lemma 17 (Fair construction). *Suppose that a* $\mathit{PresPred}^C_S$*-proof for the sequent* $\Gamma \vdash \Delta \Downarrow C$ *exists. Every fair* $\mathit{PresPred}^C$*-proof of* $\Gamma \vdash \Delta \Downarrow ?$ *whose root is annotated with the set* U *generates a constraint* D *with* $\forall U.C \Rightarrow \forall U.D$*.*

Proof on page 211

Intuitively, this means that every fair proof Q of a provable sequent $\Gamma \vdash \Delta \Downarrow ?$ contains a finite proof Q' of the sequent $\Gamma \vdash \Delta \Downarrow C$ for some valid constraint C (applications of CLOSE have be added to close Q', of course). Moreover, because of Lem. 11, it can be observed that every closed exhaustive proof of $\Gamma \vdash \Delta \Downarrow ?$ that contains Q' as an initial part has a valid constraint. This implies the completeness of proof construction with fair rule, formula, and branch selection.

7 Weakening to Eliminate Irrelevant Formulae

The calculus PresPred^C allows to ignore unneeded formulae when the rule CLOSE is applied, which is used in Sect. 5.1 and 5.2 by selecting only those formulae that do not contain U-constants. Leaving out the formulae that contain U-constants is important for two reasons: it is required for the quantifier elimination lemma (Lem. 15), but it also helps to keep constraints as small as possible. Concerning the latter argument, the precision of the U-criterion can be improved by eliminating irrelevant formulae as early as possible instead of waiting until CLOSE is applied. Since the conditions for exhaustive proofs in Sect. 5.1 require that the set U is reset to $\emptyset$ whenever the rules EX-RIGHT(-D) and ALL-LEFT(-D) occur, it can otherwise happen that formulae that were at some point identified as unnecessary can later in the proof again be considered relevant.

The classical weakening rule for a sequent calculus can directly be carried over to constrained sequents and is sound:

$$\frac{\Gamma \vdash \Delta \Downarrow C}{\Gamma, \Gamma' \vdash \Delta', \Delta \Downarrow C} \text{ WEAKEN}$$

The application of this rule has to be restricted, however, so that Lem. 11 (constraint completeness) is preserved. In the style of the conditions given in Sect. 5.1, we can assume that the conclusion and the premiss of an application of WEAKEN are both annotated with a set U of constants (in principle, one could also choose different sets for the premiss and the conclusion, but this would not lead to any interesting generalisations at this point). A sufficient condition to preserve Lem. 11 is:

$$\forall U.\ (\Gamma_p, \Gamma'_p \vdash \Delta'_p, \Delta_p) \ \Rightarrow\ \forall U.\ (\Gamma_p \vdash \Delta_p)$$

Two possible criteria that both ensure this implication are:

- *Elimination of antecedent equations:* $\Gamma' = \{c + t \doteq 0\}, \Delta' = \emptyset$, where $c \in U$ is a constant that does not occur in Γ, Δ.
- *Elimination of a group of satisfiable literals:* in certain cases, a group of inequalities, inequations and divisibility judgements can simultaneously be eliminated:

  $$\Gamma' \ =\ \{t_i \dot{\geq} 0\}_i \cup \{t'_j \dot{\leq} 0\}_j, \quad \Delta' \ =\ \{s_k \doteq 0\}_k \cup \{\alpha_l \mid u_l\}_l$$

  This is possible if the invalidity of the literals is ensured through a constant $c \in U$ such that:
  - no formula in Γ, Δ contains c;
  - Γ' contains only lower or only upper bounds on c, i.e., c occurs in each t_i with a positive coefficient and in each t'_j with a negative coefficient, or vice versa;
  - c occurs in each s_k with a non-zero coefficient;
  - c occurs in each u_l of a divisibility judgement $\alpha_l \mid u_l$ with the non-zero coefficient β_l, and:

    $$\sum_l \frac{|\gcd(\alpha_l, \beta_l)|}{|\alpha_l|} < 1 \tag{2}$$

To understand the last requirement, note that the integers (values of c) that satisfy a judgement $\alpha \mid (\beta c + t)$, provided that there are any, are periodical with the following period:

$$\frac{|\operatorname{lcm}(\alpha, \beta)|}{|\beta|} = \frac{|\alpha|}{|\gcd(\alpha, \beta)|}$$

The inequality (2) ensures that there are values for c such that none of the divisibility judgements holds (equivalently, there are infinitely many such values).

8 Refined Constraint Propagation

All calculi that we have defined so far have a severe disadvantage compared to normal FOL calculi: there is no notion of "non-unifiability," because the rule PRED-UNIFY can be applied in a very unrestricted manner to arbitrary pairs of literals that start with the same predicate symbol. This can lead to constraints that contain redundant information and to unnecessary proof splitting. For instance, in the following proof the rule PRED-UNIFY is applicable and introduces a conjunction that can lead to a splitting of the branch:

$$\dfrac{\dfrac{\dfrac{p(c,c) \ \vdash\ c - d \doteq 0 \land c - e \doteq 0, p(d,e) \ \Downarrow ?}{p(c,c) \ \vdash\ p(d,e) \ \Downarrow ?}\ \textsc{pred-unify}}{\exists x.p(x,x) \ \vdash\ \forall x,y.p(x,y) \ \Downarrow ?}\ \textsc{ex-left, all-right}}{\cdots}$$

The conjunction $c - d \doteq 0 \land c - e \doteq 0$ describes a special case, however, and can be falsified by choosing suitable values for the universally quantified symbols c, d, e. It is therefore not helpful to select this formula when applying CLOSE.

It is not possible to exclude equations involving (only) constants that stem from universal quantifiers altogether. Intuitively, our logic lacks a Herbrand theorem that would enable us to restrict our attention to Herbrand structures, in which different constants would be interpreted with themselves and thus always have distinct values.

Example 18. A proof in which also seemingly "non-unifiable" equations are essential is the following:

$$\dfrac{\dfrac{\dfrac{\dfrac{\vdash\ a \doteq 0, b \doteq 0 \ \Downarrow ?}{\vdash\ a \doteq 0 \lor b \doteq 0 \ \Downarrow ?} \qquad \dfrac{\vdash\ a \doteq 0 \land b - 1 \doteq 0, a \not\doteq 0 \land b \doteq 0 \ \Downarrow ?}{\vdash\ a \doteq 0 \land b - 1 \doteq 0 \lor a \not\doteq 0 \land b \doteq 0 \ \Downarrow ?}}{\vdash\ (a \doteq 0 \lor b \doteq 0) \land (a \doteq 0 \land b - 1 \doteq 0 \lor a \not\doteq 0 \land b \doteq 0) \ \Downarrow ?}}{\vdash\ \exists y.(a \doteq 0 \lor y \doteq 0) \land (a \doteq 0 \land y - 1 \doteq 0 \lor a \not\doteq 0 \land y \doteq 0) \ \Downarrow ?}}{\vdash\ \forall x.\exists y.(x \doteq 0 \lor y \doteq 0) \land (x \doteq 0 \land y - 1 \doteq 0 \lor x \not\doteq 0 \land y \doteq 0) \ \Downarrow ?}$$

It can be observed that the formula in the root of the proof is valid. Although the constant a comes from the universal quantifier $\forall x$, none of the formulae $a \doteq 0$ and $a \doteq 0 \land b \doteq 0$ can be left out when applying CLOSE without invalidating the constraint resulting from the proof.

In this section, we define a global criterion that tells which formulae can be ignored when applying CLOSE. This is done by distinguishing those constants in a proof that are introduced by universal quantifiers and that do not occur in illegal positions (we will call such constants *free*). Because there is no necessity to apply rules other than CLOSE to PA formulae (according to the notion of a fair proof in Sect. 6), the application of the rule PRED-UNIFY can be skipped as well if it can be predicted that the generated conjunction is irrelevant.

As a prerequisite, we need to replace the rule DIV-CLOSE with a modified version DIV-CLOSE' that enables us to order the constants in a proof in a more fine-grained way:

$$\frac{\Gamma, \alpha c' - t \doteq 0, c - c' \doteq 0 \ \vdash\ \Delta \Downarrow C}{\Gamma, \alpha c - t \doteq 0 \ \vdash\ \Delta \Downarrow [x/t]C' \vee \alpha \nmid t}\ \text{DIV-CLOSE'}$$

where c' does not occur in the conclusion and C' is a PA formula such that $C \Leftrightarrow [x/\alpha c']C'$. The rule can essentially be used in the same way as DIV-CLOSE and is not in conflict with any other part of the article. The rule has not been introduced earlier mainly because it would have made the previous sections unnecessarily complicated (but it is, in fact, the rule that is used in the implementation of the calculus).

Everywhere in this section, assume that P is an open $\mathrm{PresPred}^C$-proof in which CLOSE is never applied. For reasons of presentation, we further assume that the constants that are introduced in P by the rules ALL-LEFT, ALL-RIGHT, etc. are all pairwise distinct (and also different from "global" constants that are not explicitly introduced by any rule), which can be achieved by renaming.

The proof P induces a strict partial order $\prec_P$ on the set of all constants occurring in P, based on the order of introduction: we define $c \prec_P d$ to hold iff c is a global constant and d is not, or if the rule application that introduces c is on the path from the root of P to the rule application that introduces d (d is introduced "after" c).

Given $\prec_P$, we say that a formula ϕ is *shielded* by a constant c if ϕ is equivalent to a formula $\alpha c + t \doteq 0 \wedge \psi$ such that $\alpha \neq 0$ and $d \prec_P c$ for all constants d that occur in t. We say that ϕ is shielded by a set of constants M if ϕ is shielded by some constant $c \in M$. This definition is chosen such that the formula $\forall c.\psi$ is equivalent to *false* if ψ is a finite disjunction of formulae that are shielded by c, and similarly for sets M. The maximality condition ensures that shieldedness is preserved by quantification over bigger constants.

We say that Q is a set of *free* constants for the proof P if there is a superset $Q_c \supseteq Q$ of constants such that the following conditions are satisfied:

- all constants in Q are universal in P, i.e., are introduced by the rules ALL-RIGHT, EX-LEFT, or COL-RED;
- whenever COL-RED-SUBST is applied and the term $c - u$ contains a constant from Q_c, then also $c' \in Q_c$;
- whenever DIV-CLOSE' is applied, the term t does not contain any constants from Q_c.

Given such sets Q, Q_c, we now consider two ways to close the proof P by applying CLOSE to each of the goals. The resulting closed proofs are called P_1, P_2, and we demand that they have the following property: whenever CLOSE is applied in P_1, then (i) the disjunction C of selected PA formulae does not contain any constants from Q_c, and (ii) the disjunction of PA formulae selected in P_2 by the corresponding application of CLOSE is equivalent to $C \vee \bigvee_{i=1}^{n} \phi_i$ such that each formula ϕ_i is shielded by Q and only contains constants that occur in the considered proof goal.

Lemma 19 (Shielded Constraints). *Let C_1 be the constraint of any proof node in P_1 and C_2 the constraint of the corresponding node in P_2. Then (i) C_1 does not contain any constants from Q_c, and (ii) C_2 is equivalent to $C_1 \vee \bigvee_{i=1}^{n} \phi_i$ where each formula ϕ_i is closed, shielded by Q, and only contains constants that are global or introduced on the path from the proof root to the location of C_2.*

Proof on page 216

It is an implication of the lemma that the constraints that arrive at the roots of P_1 and P_2 are equivalent. This means that the additional formulae that are selected in P_2 when applying CLOSE, compared to P_1, did not contribute to the constraint and could have been left out right away. In case CLOSE is applied in P_2 in the most liberal way (in each goal, all PA formulae are selected), this tells which of the formulae are irrelevant for the proof.

Example 20. We show how the criterion rules out non-unifiable pairs of literals:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{f(a,2), f(b,3) \;\vdash\; f(b,c), \ldots \;\Downarrow ? \qquad f(a,2), f(b,3) \;\vdash\; c \mathrel{\dot{\geq}} 0, \ldots \;\Downarrow ?}{f(a,2), f(b,3) \;\vdash\; f(b,c) \land c \mathrel{\dot{\geq}} 0, \ldots \;\Downarrow ?}\,\text{AND-RIGHT}}{f(a,2), f(b,3) \;\vdash\; \exists z.(f(y,z) \land z \mathrel{\dot{\geq}} 0) \;\Downarrow ?}\,\text{EX-RIGHT}}{f(a,2) \land f(b,3) \;\vdash\; \exists z.(f(y,z) \land z \mathrel{\dot{\geq}} 0) \;\Downarrow ?}\,\text{AND-LEFT}}{\vdash\; f(a,2) \land f(b,3) \to \exists z.(f(y,z) \land z \mathrel{\dot{\geq}} 0) \;\Downarrow ?}\,\text{OR-RIGHT, NOT-RIGHT}$$

$$\dfrac{\vdash\; f(a,2) \land f(b,3) \to \exists z.(f(y,z) \land z \mathrel{\dot{\geq}} 0) \;\Downarrow ?}{\vdash\; \forall x, y.(f(x,2) \land f(y,3) \to \exists z.(f(y,z) \land z \mathrel{\dot{\geq}} 0) \;\Downarrow ?}\,\text{ALL-RIGHT} \times 2$$

In the left goal, the rule PRED-UNIFY can be applied to the pairs $f(a,2), f(b,c)$ and $f(b,3), f(b,c)$. Because the constants $Q = Q_c = \{a, b\}$ are free, the first pair can be ignored as it would generate the formula $b - a \doteq 0 \land c - 2 \doteq 0$ that is shielded by b in the first equation.

In the proof in Example 18, the formula $a \not\doteq 0 \land b \doteq 0$ is only shielded by the constant b that comes from an existential quantifier. This means that if the formula is to be selected for CLOSE, neither can a be a free constant, and thus none of the formulae $a \doteq 0$ and $a \doteq 0 \land b \doteq 0$ is shielded either.

There are several ways to generalise the approach described in this section:

- It is possible to vary the definition of "shielded formulae," e.g., to also consider formulae that are shielded through inequalities.
- The ordering $\prec_P$ on constants can be defined less total, again liberalising the notion of shielded formulae: if a sequence of quantifiers of the same kind is instantiated, there is no need to order the introduced constants. In Example 20, this would apply to the constants a, b.

- Sets Q of free constants can be localised, it is not necessary to use the same sets for a whole proof. It can be the case that the conditions for freeness are generally true in a proof, but are violated in a small subproof. In this situation, it might be possible to use a smaller set Q for this particular subproof. In the right subproof of the next example, for instance, c is not free because it occurs in the unshielded formula $d - c \doteq 0$. Because c does not occur in the constraint *true* of the subproof as a whole anymore, however, this is irrelevant for the rest of the proof. Consequently, it might be possible to avoid the unification of $p(c)$ and $p(e)$ in the left subproof.

$$\dfrac{p(c) \;\vdash\; p(e) \;\Downarrow ? \qquad \dfrac{\dfrac{\dfrac{*}{p(c) \;\vdash\; d - c \doteq 0, p(d), \exists x.p(x) \;\Downarrow d - c \doteq 0}}{p(c) \;\vdash\; p(d), \exists x.p(x) \;\Downarrow d - c \doteq 0}}{p(c) \;\vdash\; \exists x.p(x) \;\Downarrow \mathit{true}}}{\cdots}$$

9 Implementation and Initial Experimental Results

We have implemented the calculus defined in this paper (essentially in the version of Sect. 5) in the theorem prover Princess, which is available for download.[1] At the time of writing this section, all features introduced in the paper are implemented, apart from the optimisation of Sect. 8. In certain situations, Princess additionally uses analytic cuts [11] and formula simplification [12] to avoid redundancy in proofs. Constraints are simplified using the approach of Sect. 5.3, which means that no separate decision procedure for Presburger arithmetic is necessary. Princess is written in the Scala programming language [13] and runs on a Java virtual machine.

Most available benchmarks for SMT-solvers require uninterpreted functions or further theories like arrays that have to be handled using appropriate encodings in our calculus. Although we plan to add preprocessors for these theories to Princess, implementations of such encodings are not available yet (and also require further research in some cases). Our experimental results up to now are, therefore, restricted to the categories `QF_LIA` (quantifier-free linear integer arithmetic) and `QF_IDL` (quantifier-free integer difference logic) of the SMT library [14], see Fig. 3.

Although Princess is not primarily designed for the problems in the tested categories (in contrast to SMT-solvers), the results are reasonably good. Unsurprisingly, Princess performs better for problems that focus on arithmetic (like CIRC) than for problems that are essentially combinatoric (like queens_bench), for which more advanced techniques developed for SAT- and SMT-solvers are necessary. This fact might also explain the poor results for the nec-smt directories. In the SMT competition in 2007,[2] Princess would have solved 109 out of 203 selected problems in `QF_LIA` and 85 out of 203 problems in `QF_IDL`. For 2008, the

[1] `http://www.cs.chalmers.se/~philipp/princess`

[2] `http://www.smtcomp.org/`

QF_LIA			
Directory	*solved/total*	*Directory*	*solved/total*
Averest	10/ 19	nec-smt/small	17/ 35
CIRC	33/ 51	nec-smt/med	3/ 364
RTCL	2/ 2	nec-smt/large	0/2381
check	5/ 5	rings	53/ 293
mathsat	55/121	wisa	4/ 5
QF_IDL			
Averest	195/252	planning	5/ 45
cellar	0/ 14	qlock	0/ 72
check	3/ 3	queens_bench	53/ 297
diamonds	18/ 36	RTCL	30/ 33
DTP	0/ 60	sal	27/ 50
mathsat	96/146	sep	17/ 17
parity	34/248		

Fig. 3. Princess statistics for the categories `QF_LIA` and `QF_IDL` of the SMT library [14] (Dual Core AMD Opteron 270 with 2GHz, 1.5GB of heapspace, timeout of 1000s). Detailed results are available at `http://www.cs.chalmers.se/~philipp/princess`.

result drops to only 10 out of 205 problems in `QF_LIA` and 5 out of 203 in `QF_IDL`, presumably because of the poor performance for combinatoric problems and the lack of lemma learning (`QF_LIA` is in 2008 dominated by nec-smt problems).

10 Related Work

Model evolution modulo linear integer arithmetic [7] is a recently proposed variant of the Model Evolution calculus that is similar to our calculus in that it supports PA enhanced with uninterpreted predicates (and without functions) as input language, and that its architecture resembles tableau calculi. Model Evolution does not use rigid free variables that are shared among different branches in the way tableaux do, however, which means that also constraints can be kept branch-local. Further differences are that $\mathcal{ME}$(LIA) works on clauses, only supports a restricted form of existential quantification, and has a more explicit representation of candidate models.

SMT-solvers based on the DPLL(T) architecture [15] can handle ground problems modulo integer arithmetic (and many other theories) efficiently, but only offer heuristic quantifier handling. Because of the similarity between DPLL and sequent calculi, the work presented in this paper can be seen as an alternative approach to handling quantifiers that should also be applicable to DPLL(T).

Our approach has similarities with the framework in [16] for integrating theories into tableau calculi by distinguishing between a foreground reasoner (handling FOL) and a background reasoner (handling the theory). According to this nomenclature, the rules in Fig. 2 implement the (partial) background reasoner. Because our theory rules operate destructively on sequents, we integrate background and foreground reasoning more closely than proposed in [16]. The biggest difference between our approach and [16] is that no theory unification is performed in our calculus, it is only necessary to check the validity of constraints.

An approach to embed algebraic constraints in tableau calculi is described in [17], where quantifier elimination tasks in real arithmetic (possibly involving more than one proof goal) are carried out by an external procedure, in a manner comparable to the simultaneous solving of constraints from multiple proof goals described here. Uninterpreted functions or predicates are not handled.

There are a number of approaches to include theories into resolution-based calculi. [18] works with constraints that are solved in a theory, but requires to enumerate the solutions of constraints (whereas it is enough to check the validity of constraints in our work). In [19], while it is enough to check satisfiability of constraints, no uninterpreted functions or predicates are supported. A recent calculus to handle rational arithmetic is given in [20], and is similar to our work in that it has built-in rules to solve systems of equations and inequalities (based on Fourier-Motzkin). The calculus is complete under restrictions that effectively prevent quantification over rationals. It remains to be investigated how this fragment is related to the fragments discussed here.

11 Conclusions and Future Work

We have presented a novel calculus to reason about problems in first-order logic modulo linear integer arithmetic. As main results, we have shown that the calculus is complete for first-order logic, can decide PA, is at least as complete as the calculus $\mathcal{ME}$(LIA), and allows fair construction of proofs. We have also described refinements of the calculus and given experimental results.

Apart from continuing the implementation and further benchmarks, there are a number of concepts that require more research, among others: the encoding and handling of functions and further theories; the integration of lemma learning; the integration of connectivity conditions to make proof search more directed; the elimination of cuts in proofs. We also plan to extend our calculus to support nonlinear arithmetic (following the work in [8]), and possibly rational arithmetic.

References

1. Giese, M.: Incremental closure of free variable tableaux. In Goré, R., Leitsch, A., Nipkow, T., eds.: Proceedings, IJCAR, Siena, Italy. Volume 2083 of LNAI., Springer (2001) 545–560
2. Pugh, W.: The Omega test: a fast and practical integer programming algorithm for dependence analysis. In: Proceedings, 1991 ACM/IEEE conference on Supercomputing, New York, NY, USA, ACM (1991) 4–13
3. Presburger, M.: Über die Vollständigkeit eines gewissen Systems der Arithmetik ganzer Zahlen, in welchem die Addition als einzige Operation hervortritt. In: Sprawozdanie z I Kongresu metematyków slowiańskich, Warszawa 1929, Warsaw, Poland (1930) 92–101,395

4. Fitting, M.C.: First-Order Logic and Automated Theorem Proving. 2nd edn. Springer-Verlag, New York (1996)
5. Dershowitz, N., Manna, Z.: Proving termination with multiset orderings. Commun. ACM **22** (1979) 465–476
6. Cooper, D.: Theorem proving in arithmetic without multiplication. Machine Intelligence **7** (1972) 91–99
7. Baumgartner, P., Fuchs, A., Tinelli, C.: Melia – model evolution with linear integer arithmetic constraints. Submitted (2008)
8. Rümmer, P.: A sequent calculus for integer arithmetic with counterexample generation. In Beckert, B., ed.: Proceedings, 4th International Verification Workshop. Volume 259 of CEUR (`http://ceur-ws.org/`). (2007)
9. Schrijver, A.: Theory of Linear and Integer Programming. Wiley (1986)
10. Norrish, M.: Complete integer decision procedures as derived rules in HOL. In: Proceedings, Theorem Proving in Higher Order Logics. Volume 2758 of LNCS., Springer (2003) 71–86
11. D'Agostino, M.: Tableaux methods for classical propositional logic. In D'Agostino, M., Gabbay, D., Hähnle, R., Posegga, J., eds.: Handbook of Tableau Methods. Kluwer, Dordrecht (1999) 45–123
12. Massacci, F.: Simplification: A general constraint propagation technique for propositional and modal tableaux. In de Swart, H., ed.: Proceedings, International Conference on Automated Reasoning with Analytic Tableaux and Related Methods, Oosterwijk, The Netherlands. Volume 1397 of LNAI., Springer (1998) 217–232
13. Odersky, M., Spoon, L., Venners, B.: Programming in Scala. Artima Inc., Mountain View, CA, USA (2008) to appear.
14. Barrett, C., Ranise, S., Stump, A., Tinelli, C.: The Satisfiability Modulo Theories Library (SMT-LIB). `http://www.smt-lib.org` (2008)
15. Nieuwenhuis, R., Oliveras, A., Tinelli, C.: Solving SAT and SAT modulo theories: From an abstract Davis-Putnam-Logemann-Loveland procedure to DPLL(T). Journal of the ACM **53** (2006) 937–977
16. Beckert, B.: Equality and other theories. In D'Agostino, M., Gabbay, D., Hähnle, R., Posegga, J., eds.: Handbook of Tableau Methods. Kluwer, Dordrecht (1999) 197–254
17. Platzer, A.: Differential dynamic logic for verifying parametric hybrid systems. In Olivetti, N., ed.: Proceedings, International Conference on Automated Reasoning with Analytic Tableaux and Related Methods, Aix en Provence, France. Volume 4548 of LNCS., Springer (2007) 216–232
18. Stickel, M.E.: Automated deduction by theory resolution. Journal of Automated Reasoning **1** (1985) 333–355
19. Bürckert, H.J.: A resolution principle for clauses with constraints. In Stickel, M.E., ed.: Proceedings, 10th International Conference on Automated Deduction. Volume 449 of LNCS., Springer (1990) 178–192
20. Korovin, K., Voronkov, A.: Integrating linear arithmetic into superposition calculus. In Duparc, J., Henzinger, T.A., eds.: Proceedings, 21st International Workshop on Computer Science Logic. Volume 4646 of LNCS., Springer (2007) 223–237
21. Hintikka, J., Sandu, G.: Game-theoretical semantics. In Benthem, J.F.V., Meulen, A.G.T., eds.: Handbook of Logic and Language. MIT Press, Cambridge, Massachusetts (1997)
22. Gale, D., Stewart, F.M.: Infinite games with perfect information. In Kuhn, H.W., Tucker, A.W., eds.: Contributions to the Theory of Games II. Volume 28 of Annals of Mathematics Studies. Princeton University Press, Princeton NJ (1953) 245–266

A Proofs

Lemma 2 (Soundness of Pred^C)

All rules of the calculus Pred^C are sound in the sense introduced in Sect. 2 (but considering evaluation over arbitrary first-order structures (U, I) with an arbitrary non-empty universe U). Some of the cases are:

- AND-RIGHT: Assume $\Gamma \vdash \phi \wedge \psi, \Delta \Downarrow C \wedge D$ is invalid, i.e., for some structure (U, I) and some constant assignment δ we have $val_{(U,I),\delta}(C \wedge D) = tt$ but $val_{(U,I),\delta}(\Gamma \vdash \phi \wedge \psi, \Delta) = ff$. This implies:
  - $val_{(U,I),\delta}(C) = tt$ and $val_{(U,I),\delta}(D) = tt$,
  - $val_{(U,I),\delta}(\phi) = ff$ or $val_{(U,I),\delta}(\psi) = ff$.

  Then $val_{(U,I),\delta}(\Gamma \vdash \phi, \Delta \Downarrow C) = ff$ or $val_{(U,I),\delta}(\Gamma \vdash \psi, \Delta \Downarrow D) = ff$ and one of the premisses has to be invalid.
- OR-RIGHT: Assume that $\Gamma \vdash \phi \vee \psi, \Delta \Downarrow C$ is invalid, i.e., for some structure (U, I) and some constant assignment δ we have $val_{(U,I),\delta}(C) = tt$ but $val_{(U,I),\delta}(\Gamma \vdash \phi \vee \psi, \Delta) = ff$. This implies:
  - $val_{(U,I),\delta}(\phi) = ff$ and $val_{(U,I),\delta}(\psi) = ff$.

  Then also $val_{(U,I),\delta}(\Gamma \vdash \phi, \psi, \Delta \Downarrow C) = ff$ and the premiss is invalid.
- ALL-RIGHT: Assume that $\Gamma \vdash \forall x.\phi, \Delta \Downarrow \forall x.C$ is invalid, i.e., for some structure (U, I) and some constant assignment δ we have $val_{(U,I),\delta}(\forall x.C) = tt$ but $val_{(U,I),\delta}(\Gamma \vdash \forall x.\phi, \Delta) = ff$. Because c is a fresh constant, this implies that there is an assignment δ' that agrees with δ on all constants but c such that $val_{(U,I),\delta'}([x/c]\phi) = ff$. For this δ', also $val_{(U,I),\delta'}([x/c]C) = tt$ holds. Then $val_{(U,I),\delta'}(\Gamma \vdash [x/c]\phi, \Delta \Downarrow [x/c]C) = ff$ and therefore the premiss is invalid.
- EX-RIGHT: Assume that $\Gamma \vdash \exists x.\phi, \Delta \Downarrow \exists x.C$ is invalid, i.e., for some structure (U, I) and some constant assignment δ we have $val_{(U,I),\delta}(\exists x.C) = tt$ but $val_{(U,I),\delta}(\Gamma \vdash \exists x.\phi, \Delta) = ff$. Because c is a fresh constant, this implies that there is an assignment δ' that agrees with δ on all constants but c such that $val_{(U,I),\delta'}([x/c]C) = tt$ and $val_{(U,I),\delta'}([x/c]\phi) = ff$. Then $val_{(U,I),\delta'}(\Gamma \vdash [x/c]\phi, \exists x.\phi, \Delta \Downarrow [x/c]C) = ff$ and the premiss is invalid.

The conjecture follows by a simple induction on the size of proofs.

Lemma 3 (Completeness of Pred^C)

We assume that $\vdash \phi \Downarrow C$ is unprovable for all valid formulae C and deduce that ϕ is not valid. The main difficulty with the approach is to use the knowledge

> "All constraints that can be derived during a proof attempt are invalid"

to identify a saturated proof branch from which a countermodel can be constructed using the normal Hintikka-construction [4]. We bridge this gap by introducing a notion of "most general constraints" (mgc), which are basically constraints that are the disjunction of all closing constraints that can be derived from a proof. Because saturated proofs are usually infinite, also mgc can be infinitary. Importantly, we can show that the mgc is valid iff there is a valid closing constraint. From the fact that the mgc is invalid (if a formula is unprovable), we can derive a saturated proof branch that we turn into a countermodel.

To derive most general constraints, we modify the closure rules of Pred^C and allow to introduce constraints with disjunctions. The calculus Pred^C_J consists of the same rules as Pred^C apart from PRED-CLOSE, which is replaced with the two following rules:

$$\frac{\Gamma, p(s_1, \ldots, s_n) \;\vdash\; p(t_1, \ldots, t_n), \Delta \;\Downarrow C}{\Gamma, p(s_1, \ldots, s_n) \;\vdash\; p(t_1, \ldots, t_n), \Delta \;\Downarrow C \vee \bigwedge_i s_i \doteq t_i} \;\textsc{disj-close}$$

$$\frac{*}{\Gamma \;\vdash\; \Delta \Downarrow \mathit{false}} \;\textsc{false-close}$$

We call a (possibly infinite) proof tree in Pred^C_J *fair* if

- all structural/propositional rules and skolemisation are eventually applied whenever they are applicable,
- the rule DISJ-CLOSE is applied infinitely often to each complementary pair on each branch,
- the rule ALL-LEFT/EX-RIGHT is applied infinitely often to every universally/existentially quantified formula in the antecedent/succedent on each branch,
- the rule FALSE-CLOSE is only applied if no other rule is applicable.

The constraints generated by fair proof trees are called *most general constraints*. Such constraints can be infinitary and contain infinitely many quantifiers, disjunctions (due to infinite branches in a proof tree) or conjunctions (due to infinitely many branches in a proof tree). We consider infinitary formulae as infinite trees, in which conjunctions and disjunctions are always seen as binary connectives. Because no function symbols are involved, terms are always single variables or constants, and literals occurring in the formulae are always finite. We use game semantics (cf. [21]) to give meaning to infinitary formulae.

Infinitary Formulae

Starting with a fixed non-empty domain U, initial variable/constant assignments β, δ and at the root of a formula, two players (the *verifier* and the *falsifier*) play against each other:

- the verifier tries to show that the formula is true. When arriving at $\vee$ or $\exists x$, the verifier has to choose the subformula to continue with, or the value that x is to be given.
- the falsifier tries to show that the formula is false. When arriving at $\wedge$ or $\forall x$, the falsifier has to choose the subformula to continue with, or the the value that x is to be given.

The verifier wins if the game ends and arrives at a literal that evaluates to tt. The falsifier wins if the game does not terminate,[3] or if it ends at a literal that evaluates to ff. A formula is true (in a certain structure) if the verifier has a (deterministic) winning strategy, i.e., the verifier can win whatever the falsifier does, and false otherwise. By the theorem of Gale-Stewart [22], in the second case the falsifier has a (deterministic) winning strategy. A formula is valid if the verifier has winning strategies for each domain U and assignments β, δ.

It is well-known that the above game semantics coincides with Tarski semantics for finite formulae.

Properties of Most-General Constraints

The important property of the most-general constraint obtained from a fair Pred^C_J-proof is that it subsumes all constraints that the original calculus Pred^C could possibly have generated. By *possibly generated constraints* we mean all constraints that can be obtained from the Pred^C_J-proof by turning the proof into a Pred^C-proof:

- remove all applications of FALSE-CLOSE and DISJ-CLOSE,
- chop off all infinite branches at some point to make the proof tree finite, and
- apply PRED-CLOSE in some way to all open branches.

The constraint C that arrives at the root $\Gamma \vdash \Delta \Downarrow C$ of the resulting proof is called *possible generated.*

Lemma 21. *The most-general constraint of a fair Pred^C_J-proof is valid iff the proof possibly generates a valid constraint.*

Assume that all possibly generated constraints of a fair Pred^C_J-proof are invalid. By the lemma, this means that the mgc of this proof is invalid: for some particular domain, the falsifier has a winning strategy for the mgc. We can then discover the right branch in the proof tree and simultaneously construct a countermodel based on the above domain by playing a game (that the falsifier wins).

Proof (Lem. 21).

"$\Longleftarrow$:" Assume that a fair Pred^C_J-proof possibly generates a valid constraint. We fix a domain U, variable/constant assignments β, δ, and a winning strategy S_1 for the verifier for this valid constraint (note, that S_1 is only responsible for nodes $\exists x$). We can derive a winning strategy S_2 for the verifier for the mgc:

- Initially, S_1 and S_2 behave in the same way when arriving at a node $\exists x$. When arriving at $\vee$ in the mgc, which has to be introduced by DISJ-CLOSE, S_2 chooses the left branch (and not the conjunction $\bigwedge_i s_i \doteq t_i$).
- Once the game has reached a point where the rule PRED-CLOSE was applied to produce a conjunction ϕ in the possibly generated constraint, S_2 changes its behaviour:

[3] Note, that this implies that $\mathit{true} \wedge \mathit{true} \wedge \cdots$ is false, which is the intended semantics.

- when arriving at a node $\exists x$, the value of x is chosen arbitrarily;
- when arriving at $\vee$, which has to be introduced by DISJ-CLOSE, S_2 chooses the left branch if it leads to the conjunction ϕ, otherwise the right branch.

Because of fairness, DISJ-CLOSE is eventually applied to every complementary pair on every branch, so that S_2 is guaranteed to win after a finite number of steps.

"$\Longrightarrow$:" Assume that the mgc is valid. Further, assume that all variables that are bound in the mgc are pairwise distinct, and that for each variable x the symbol f_x denotes a function symbol whose arity equals the number of variables that are bound above the location where x is bound. Finally, let U be the domain of the free term algebra over the vocabulary consisting of the functions f_x (we possibly have to add further functions that do not belong to any variables to ensure that U is non-empty).

We fix U as the domain of evaluation (and arbitrary assignments β, δ) as well as a winning verifier strategy S. For each branch b in the proof tree, we construct a falsifier strategy T_b:

- When arriving at a quantifier $\forall x$, the falsifier chooses $f_x(\beta(y_1), \dots, \beta(y_k))$ as the value of x, where $\beta(y_1)$, …, $\beta(y_k)$ are the values given to variables bound above $\forall x$.
- When arriving at $\wedge$ that was introduced by AND-RIGHT or OR-LEFT, the falsifier follows the branch b.
- When arriving at other $\wedge$, the falsifier chooses an arbitrary branch.

Because S is a winning strategy, the verifier wins against each of the strategies T_b in a finite number of steps, which means that S picks one particular application of DISJ-CLOSE on each branch. Because S is deterministic and the strategies T_b only differ in the treatment of $\wedge$, no two selected DISJ-CLOSE applications are located on the same branch. This implies that the selected DISJ-CLOSE applications can be replaced with PRED-CLOSE to turn the fair Pred_J^C-proof into a Pred^C-proof (removing all other applications of DISJ-CLOSE and FALSE-CLOSE).

The constraint of the resulting Pred^C-proof is valid: because U is the domain of a term algebra, the values chosen by the verifier for the existentially quantified variables describe a simultaneous unifier of the equations produced by PRED-CLOSE.

Selection of the right branch

We will now show how the reasoning of the previous pages can be used to detect the right saturated branch in a proof tree and to construct a countermodel of the formulae on this branch. To this end, assume that all possibly generated constraints of a fair Pred_J^C-proof are invalid. By Lem. 21, this means that the mgc of this proof is invalid: for some particular domain, the falsifier has a winning strategy for the mgc (note, that the mgc only contains equations between variables as atoms). Wlog, we may assume that the domain is countable and consists of the elements $a_1, a_2, \ldots$ (this follows by the same argument as in the proof of Lem. 21: if no such countable domain would exist, we could construct a countable term algebra for which the verifier has a winning strategy and conclude the validity of the mgc).

Assume that the constants that the rules EX-RIGHT, ALL-LEFT introduce in the Pred_J^C-proof are all pairwise distinct. We now discover the right branch in the proof tree and simultaneously construct a countermodel based on the above domain by playing a game. The falsifier will use its winning strategy, whereas we assume that the verifier behaves as follows:

- When arriving at $\vee$ in the mgc, which has to be introduced by DISJ-CLOSE, the verifier chooses the left branch (and not the conjunction $\bigwedge_i s_i \doteq t_i$);
- when arriving at quantifiers $\exists x$ in the mgc, i.e., at a quantified formula $\exists x.\phi$ (succedent) or $\forall x.\phi$ (antecedent) in the proof that is instantiated by EX-RIGHT or ALL-LEFT, the verifier chooses a domain element a_i as value of x. The value a_i is taken when the formula $\exists x.\phi$ or $\forall x.\phi$ is visited the i-time in the game, which means that all domain elements are systematically enumerated for each formula $\exists x.\phi$ or $\forall x.\phi$.

The path chosen by the game corresponds to one branch $S_0, S_1, \ldots$ (a sequence of sequents) in the proof tree.

Countermodel Construction. In order to construct a countermodel, we first define the notion of *persistent formulae* on the selected branch. By

$$\mathit{Lit}(\phi_1, \ldots, \phi_n \ \vdash\ \psi_1, \ldots, \psi_m \ \Downarrow ?) \ := \ \{\neg\phi_1, \ldots, \neg\phi_n, \psi_1, \ldots, \psi_m\}$$

we denote the set of literals represented by a sequent. The set of *persistent formulae* of a sequence of sequents is then defined as

$$PF \ := \ PF(S_0, S_1, \ldots) \ := \ \bigcup_i \bigcap_{j \geq i} \mathit{Lit}(S_j)$$

For predicate calculus, there are two kinds of formulae that can be persistent: existentially quantified formulae (which have to be instantiated multiple times and never disappear from a branch) and literals (to which no further rules apart from closure rules can be applied).

It is now simple to find a countermodel of all persistent atoms:

- We choose the same domain as for the game that was played in the previous section.
- We interpret constants with the values that were chosen by the verifier and the falsifier during the game. Because we assumed that the constants that are introduced by EX-RIGHT, ALL-LEFT are pairwise distinct, this yields a consistent valuation.
- We evaluate all persistent literals $p(\bar{t})$ with *ff* and all persistent literals $\neg p(\bar{t})$ with *ff* as well (i.e., $p(\bar{t})$ with *tt*). This is consistent, because whenever possibly conflicting literals $p(\bar{t})$ and $\neg p(\bar{s})$ are both persistent, we know that

DISJ-CLOSE has been applied to the pair and that the formula $\bigwedge_i s_i \doteq t_i$ evaluates to *ff* for the chosen valuation of constants (otherwise, the verifier could have won the game, which contradicts the assumption that the falsifier has a winning strategy).

To show that the chosen structure is a countermodel of *all* formulae on the proof branch, we perform the usual Hintikka-style induction on the size of formulae. The quantifier cases are the most interesting ones:

- $\forall x.\phi$ in the succedent: we know that ALL-RIGHT has been applied to the formula, and that $[x/c]\phi$ evaluates to *ff* for some constant c. Then also the quantified formula evaluates to *ff*.
- $\exists x.\phi$ in the succedent: EX-RIGHT has been applied infinitely often to the formula, and by the choice of the verifier the values of the introduced constants $c_1, c_2, \ldots$ enumerate all domain elements. Because all formulae $[x/c_1]\phi$, $[x/c_2]\phi$, $\ldots$ are known to evaluate to *ff*, also $\exists x.\phi$ evaluates to *ff*.

If we have found a countermodel for all sequents on a proof branch, then also the root of the proof tree is invalid:

Lemma 22. *If all possibly generated constraints of a fair* Pred_J^C*-proof for the sequent* $\Gamma \vdash \Delta \Downarrow ?$ *are invalid, then the root* $\Gamma \vdash \Delta$ *of the proof is invalid.*

This implies Lem. 3.

To see that also Lem. 4 holds, observe that every partial proof of $\vdash \phi \Downarrow ?$ (as described in the lemma) can be extended to a fair Pred_J^C-proof. By Lem. 22 and because ϕ is valid, this implies that some possibly generated constraint is valid as well. Because atoms are always persistent in Pred_J^C, this constraint is also generated by a finite extension of the original Pred^C-proof.

Lemma 6 (Universal Completeness of $\mathbf{PresPred}_S^C$)

Exhaustive application of all rules apart from ALL-LEFT, EX-RIGHT and CLOSE terminates. Subsequently, apply CLOSE on each goal as liberally as possible, selecting all PA formulae in the goal (the literals containing uninterpreted predicates have to be left out). If the resulting constraint C (for the whole proof tree) is not valid, a countermodel can be constructed as follows:

- Because C is not valid, there has to be an assignment δ of the constants introduced when applying EX-LEFT, ALL-RIGHT such that the constraint extracted from one of the proof goals evaluates to *ff*. Denote this proof goal by $\Gamma' \vdash \Delta' \Downarrow D$.
- Because PRED-UNIFY has been applied exhaustively in the proof, for any complementary pair $p(\bar{t}) \in \Gamma'$, $p(\bar{s}) \in \Delta'$ the argument vectors evaluate to different integer vectors given the constant assignment δ. This means that a consistent interpretation I of the predicates can be constructed from $\Gamma' \vdash \Delta' \Downarrow D$.
- Using the normal Hintikka construction, it can be shown that I is a countermodel of the original sequent $\Gamma \vdash \Delta$.

Lemma 7 (Existential Completeness of PresPred^C_S)

To prove this, we first need a further lemma:

Lemma 23 (Constant Substitution in Proofs). *Suppose $\Gamma \vdash \Delta$ is a sequent, $\sigma = [c_1/\alpha_1, \dots, c_n/\alpha_n]$ a substitution that replaces constants with integer literals, and C a constraint, such that $\sigma(\Gamma) \vdash \sigma(\Delta) \Downarrow C$ has a proof in the calculus PresPred^C_S. Then there is a constraint D such that (i) $\Gamma \vdash \Delta \Downarrow D$ has a proof in PresPred^C_S, and (ii) the implication $C \Rightarrow \sigma(D)$ holds.*

Proof. By induction on the size of the proof of $\sigma(\Gamma) \vdash \sigma(\Delta) \Downarrow C$. We can first observe that the existence of a proof for $\Gamma \vdash \Delta \Downarrow C$ implies the existence of proofs for $\Gamma, \Gamma' \vdash \Delta', \Delta \Downarrow C$. Some of the cases are then:

- The last rule applied in the proof is CLOSE:

  $$\dfrac{*}{\sigma(\Gamma, \phi_1, \dots, \phi_n) \vdash \sigma(\psi_1, \dots, \psi_m, \Delta) \Downarrow \sigma(\neg\phi_1 \vee \cdots \vee \neg\phi_n \vee \psi_1 \vee \cdots \vee \psi_m)}\ \textsc{c.}$$

  The non-ground proof can then simply be constructed as:

  $$\dfrac{*}{\Gamma, \phi_1, \dots, \phi_n \vdash \psi_1, \dots, \psi_m, \Delta \Downarrow \neg\phi_1 \vee \cdots \vee \neg\phi_n \vee \psi_1 \vee \cdots \vee \psi_m}\ \textsc{close}$$

- The last rule applied in the proof is NOT-RIGHT:

  $$\dfrac{\begin{array}{c}\vdots\\ \sigma(\Gamma), \sigma(\phi) \vdash \sigma(\Delta \backslash \sigma^{-1}(\{\sigma(\neg\phi)\})) \Downarrow C\end{array}}{\sigma(\Gamma) \vdash \sigma(\neg\phi), \sigma(\Delta) \Downarrow C}\ \textsc{not-right}$$

  Using the induction hypothesis, for a suitable constraint D there is a proof of $\Gamma, \phi \vdash \Delta \backslash \sigma^{-1}(\{\sigma(\neg\phi)\}) \Downarrow D$, therefore also of $\Gamma, \phi \vdash \Delta \Downarrow D$, which can be continued as follows:

  $$\dfrac{\begin{array}{c}\vdots\\ \Gamma, \phi \vdash \Delta \Downarrow D\end{array}}{\Gamma \vdash \neg\phi, \Delta \Downarrow D}\ \textsc{not-right}$$

- The last rule applied in the proof is EX-RIGHT:

  $$\dfrac{\begin{array}{c}\vdots\\ \sigma(\Gamma) \vdash \sigma([x/c]\phi), \sigma(\exists x.\phi), \sigma(\Delta) \Downarrow [x/c]C\end{array}}{\sigma(\Gamma) \vdash \sigma(\exists x.\phi), \sigma(\Delta) \Downarrow \exists x.C}\ \textsc{ex-right}$$

  Again, for some constraint D such that $[x/c]C \Rightarrow \sigma(D)$ there is a proof of $\Gamma \vdash [x/c]\phi, \exists x.\phi, \Delta \Downarrow D$, and by renaming we can establish $c \notin \{c_1, \dots, c_n\}$. The proof can be continued as

  $$\dfrac{\begin{array}{c}\vdots\\ \Gamma \vdash [x/c]\phi, \exists x.\phi, \Delta \Downarrow D\end{array}}{\Gamma \vdash \exists x.\phi, \Delta \Downarrow \exists x.[c/x]D}\ \textsc{ex-right}$$

  and the implication $\exists x.C \Rightarrow \exists x.[c/x]\sigma(D) \Leftrightarrow \sigma(\exists x.[c/x]D)$ holds.

Proof (Lem. 7). As the first step, we consider a calculus $\mathrm{PresPred}^C_G$ that coincides with $\mathrm{PresPred}^C_S$, only that the rules ALL-LEFT, EX-RIGHT are replaced with "ground versions" (α ranges over integer literals):

$$\frac{\Gamma \vdash [x/\alpha]\phi, \exists x.\phi, \Delta \Downarrow C}{\Gamma \vdash \exists x.\phi, \Delta \Downarrow C} \text{ EX-RIGHT-G}$$

$$\frac{\Gamma, [x/\alpha]\phi, \forall x.\phi \vdash \Delta \Downarrow C}{\Gamma, \forall x.\phi \vdash \Delta \Downarrow C} \text{ ALL-LEFT-G}$$

The valid sequent $\Gamma \vdash \Delta$ from the lemma has a proof with valid constraint in $\mathrm{PresPred}^C_G$: otherwise, construct a (possibly infinite) proof tree in which every quantified formulae on every branch has been instantiated with every integer literal. Using the normal Hintikka construction, a countermodel of $\Gamma \vdash \Delta$ can be found.

By induction on the size of the $\mathrm{PresPred}^C_G$-proof, we can transform the proof into a $\mathrm{PresPred}^C_S$-proof. Most steps in the proof are left untouched, the only changes are applied to EX-RIGHT-G, ALL-LEFT-G. For the first case (the latter case is similar), suppose the ground proof ends with:

$$\frac{\Gamma \vdash [x/\alpha]\phi, \exists x.\phi, \Delta \Downarrow C}{\Gamma \vdash \exists x.\phi, \Delta \Downarrow C} \text{ EX-RIGHT-G}$$

where C is valid. We replace the application of EX-RIGHT-G with EX-RIGHT:

$$\frac{\Gamma \vdash [x/c]\phi, \exists x.\phi, \Delta \Downarrow ?}{\Gamma \vdash \exists x.\phi, \Delta \Downarrow ?} \text{ EX-RIGHT}$$

Because $\Gamma \vdash [x/\alpha]\phi, \exists x.\phi, \Delta \Downarrow C$ has a proof in $\mathrm{PresPred}^C_S$, by Lem. 23 there is a proof of $\Gamma \vdash [x/c]\phi, \exists x.\phi, \Delta \Downarrow D$ for a suitable D such that $C \Rightarrow [c/\alpha]D$, i.e., $[c/\alpha]D$ is valid. This means that also $\exists x.[c/x]D$ is valid and the translated proof is:

$$\frac{\Gamma \vdash [x/c]\phi, \exists x.\phi, \Delta \Downarrow D}{\Gamma \vdash \exists x.\phi, \Delta \Downarrow \exists x.[c/x]D} \text{ EX-RIGHT}$$

Lemma 8 (Completeness on the $\mathcal{ME}$(LIA) fragment)

By constructing a $\mathrm{PresPred}^C_S$-proof for each solution of ϕ (with the help of Lem. 7), which can then be combined into a single proof. We first need a further lemma that allows us to restructure proofs:

Lemma 24. *Suppose a* $\mathit{PresPred}^C_S$*-proof exists for the sequent* $\Gamma \vdash \Delta \Downarrow C$*, and* $\Gamma \vdash \Delta$ *contains a formula* ϕ *to which one of the rules* AND-**,* OR-**,* NOT-**,* EX-LEFT*,* ALL-RIGHT *is applicable. For some* D *with* $C \Rightarrow D$ *there is a* $\mathit{PresPred}^C_S$*-proof of* $\Gamma \vdash \Delta \Downarrow D$ *in which the first rule application is performed on* ϕ*. The depth of the new proof (the length of the longest branch) is at most* 1 *bigger than the depth of the original proof, and the first rule application of the* *original proof is the first or the second rule application on all branches in the new proof. Further, if the original proof does not contain any rule applications to PA formulae apart from* CLOSE, *then the new proof does not contain any such applications apart from (possibly) the first rule application and* CLOSE.

Proof. Call the original proof P. The main difficulty in the proof of the lemma comes from the fact that sequents consist of sets of formulae (not of multisets), which means that multiple occurrences of a formula are implicitly contracted to only one occurrence. We therefore prove the lemma in two steps: we first show it under the assumption that antecedents and succedents in fact are multisets; as second step, it is then shown that a proof with multiset sequents can be transformed to a proof with ordinary constrained sequents.

Step 1 (proofs with multiset sequents). If P does not contain any rule application to ϕ, we simply add one as first rule application in the new proof and are finished (note, that the constraint of the proof stays equivalent). Otherwise, we show that applications of AND-*, OR-*, NOT-*, EX-LEFT, ALL-RIGHT to ϕ can be shifted towards the root in P. By an inductive argument on the size of P, assume that the second rule application on all branches of P is an application to ϕ, whereas the first rule application is done to a different formula. Note, that whenever the constraint of an inner proof node is weakened, also the constraint of the whole proof becomes weaker or does not change. The cases to be considered are:

- ϕ starts with $\neg$, with $\wedge$ and is in the antecedent, or with $\vee$ and is in the succedent. In all cases, we can simply permute the first and the second rule application in P. Because the rules that can be applied to ϕ do not affect the constraint, the overall constraint stays the same.
- ϕ starts with $\vee$ and is in the antecedent, or with $\wedge$ and is in the succedent. Again, we can permute the first and the second rule application in P, but have to argue that the constraint stays equivalent or becomes weaker. The most interesting situation is the one where the first rule application in P is ALL-LEFT or EX-RIGHT, e.g.:

$$\cfrac{\cfrac{\Gamma \vdash \ldots, \phi_1, \Delta \Downarrow [x/c]C \qquad \Gamma \vdash \ldots, \phi_2, \Delta \Downarrow [x/c]D}{\Gamma \vdash \exists x.\psi, [x/c]\psi, \phi_1 \wedge \phi_2, \Delta \Downarrow [x/c](C \wedge D)}\ \textsc{and-right}}{\Gamma \vdash \exists x.\psi, \phi_1 \wedge \phi_2, \Delta \Downarrow \exists x.(C \wedge D)}\ \textsc{ex-right}$$

  is transformed into:

$$\cfrac{\cfrac{\Gamma \vdash \ldots, [x/c]\psi, \Delta \Downarrow [x/c]C}{\Gamma \vdash \exists x.\psi, \phi_1, \Delta \Downarrow \exists x.C}\ \textsc{ex-r.} \qquad \cfrac{\Gamma \vdash \ldots, [x/c]\psi, \Delta \Downarrow [x/c]D}{\Gamma \vdash \exists x.\psi, \phi_2, \Delta \Downarrow \exists x.D}\ \textsc{ex-r.}}{\Gamma \vdash \exists x.\psi, \phi_1 \wedge \phi_2, \Delta \Downarrow \exists x.C \wedge \exists x.D}\ \textsc{and-r.}$$

  Because of $\exists x.(C \wedge D) \Rightarrow \exists x.C \wedge \exists x.D$, the resulting constraint does not becomes stronger.
- ϕ starts with $\exists$ and is in the antecedent, or with $\forall$ and is in the succedent. Again, we can permute the first and the second rule application in P. The

most interesting situation is the one where the first rule application in P is ALL-LEFT or EX-RIGHT, e.g.:

$$\dfrac{\dfrac{\Gamma \vdash \exists x.\psi, [x/c]\psi, [y/d]\phi', \Delta \Downarrow [y/d][x/c]C}{\Gamma \vdash \exists x.\psi, [x/c]\psi, \forall y.\phi', \Delta \Downarrow \forall y.[x/c]C}\ \textsc{all-right}}{\Gamma \vdash \exists x.\psi, \forall y.\phi', \Delta \Downarrow \exists x.\forall y.C}\ \textsc{ex-right}$$

is transformed into:

$$\dfrac{\dfrac{\Gamma \vdash \exists x.\psi, [x/c]\psi, [y/d]\phi', \Delta \Downarrow [y/d][x/c]C}{\Gamma \vdash \exists x.\psi, [y/d]\phi', \Delta \Downarrow \exists x.[y/d]C}\ \textsc{ex-right}}{\Gamma \vdash \exists x.\psi, \forall y.\phi', \Delta \Downarrow \forall y.\exists x.C}\ \textsc{all-right}$$

Because of $\exists x.\forall y.C \Rightarrow \forall y.\exists x.C$, the resulting constraint does not become stronger.

Step 2 (elimination of multiple occurrences of a formula). By induction on the size of a proof Q of a multiset sequent $\Gamma \vdash \phi, \phi, \Delta \Downarrow C$ with two (or more) occurrences of a formula ϕ in the succedent (or, analogously, in the antecedent), we show that: there is a proof Q' of the sequent $\Gamma \vdash \phi, \Delta \Downarrow D$ that is not bigger that Q, such that $C \Rightarrow D$. If Q does not contain any rule applications to PA formulae apart from CLOSE, then neither does Q'.

Wlog., we can assume that the first rule application in Q is done to one of the occurrences of ϕ (otherwise, consider the maximal subtrees of Q with this property, which are strictly smaller than Q and can be handled by the induction hypothesis. Outside of the maximal subtrees, no rules are applied to ϕ and the two occurrences can be replaced with only one occurrence right away). There are the following cases:

- ϕ starts with $\neg$, with $\wedge$ and is in the antecedent, or with $\vee$ and is in the succedent. By *Step 1*, we can transform Q into a proof Q_2 in which the second rule application on all branches is done to the second occurrence of ϕ (the depth of Q_2 is at most 1 bigger than that of Q). E.g.:

  $$\dfrac{\dfrac{\Gamma \vdash \phi_1, \phi_2, \phi_1, \phi_2, \Delta \Downarrow C}{\Gamma \vdash \phi_1, \phi_2, \phi_1 \vee \phi_2, \Delta \Downarrow C}\ \textsc{or-right}}{\Gamma \vdash \phi_1 \vee \phi_2, \phi_1 \vee \phi_2, \Delta \Downarrow C}\ \textsc{or-right}$$

  The induction hypothesis allows to replace the subproof for the sequent $\Gamma \vdash \phi_1, \phi_2, \phi_1, \phi_2, \Delta \Downarrow C$ with a proof of $\Gamma \vdash \phi_1, \phi_2, \Delta \Downarrow D$ with $C \Rightarrow D$ that is not bigger. Simultaneously, one of the formulae $\phi_1 \vee \phi_2$ and one of the OR-RIGHT applications can be eliminated. (Similarly for the other cases.)
- ϕ starts with $\vee$ and is in the antecedent, or with $\wedge$ and is in the succedent. By *Step 1*, we can transform Q into a proof Q_2 in which the second rule

application on all branches is done to the second occurrence of ϕ. E.g.:

$$\frac{\mathcal{A} \qquad \mathcal{B}}{\Gamma \ \vdash \ \phi_1 \wedge \phi_2, \phi_1 \wedge \phi_2, \Delta \ \Downarrow C \wedge D \wedge E \wedge F}$$

$$\cfrac{\cfrac{\Gamma \ \vdash \ \phi_1, \phi_1, \Delta \ \Downarrow C \quad \Gamma \ \vdash \ \phi_1, \phi_2, \Delta \ \Downarrow D}{\Gamma \ \vdash \ \phi_1, \phi_1 \wedge \phi_2, \Delta \ \Downarrow C \wedge D}}{\mathcal{A}}$$

$$\cfrac{\cfrac{\Gamma \ \vdash \ \phi_2, \phi_1, \Delta \ \Downarrow E \quad \Gamma \ \vdash \ \phi_2, \phi_2, \Delta \ \Downarrow F}{\Gamma \ \vdash \ \phi_2, \phi_1 \wedge \phi_2, \Delta \ \Downarrow E \wedge F}}{\mathcal{B}}$$

As before, the subproofs for $\Gamma \ \vdash \ \phi_1, \phi_1, \Delta \ \Downarrow C$ and $\Gamma \ \vdash \ \phi_2, \phi_2, \Delta \ \Downarrow F$ can be replaced with proofs of $\Gamma \ \vdash \ \phi_1, \Delta \ \Downarrow C'$ and $\Gamma \ \vdash \ \phi_2, \Delta \ \Downarrow F'$ with $C \Rightarrow C'$ and $F \Rightarrow F'$. Simultaneously, one of the formulae $\phi_1 \wedge \phi_2$ and one of the AND-RIGHT applications can be eliminated. The resulting constraint is $C' \wedge F'$ and has the property $C \wedge D \wedge E \wedge F \Rightarrow C' \wedge F'$. (Similarly for the other case.)

– ϕ starts with $\exists$ and is in the antecedent, or with $\forall$ and is in the succedent. By *Step 1*, we can transform Q into a proof Q_2 in which the second rule application on all branches is done to the second occurrence of ϕ. E.g.:

$$\cfrac{\cfrac{\Gamma \ \vdash \ [x/c]\phi', [x/d]\phi', \Delta \ \Downarrow [y/d][x/c]C}{\Gamma \ \vdash \ [x/c]\phi', \forall x.\phi', \Delta \ \Downarrow \forall y.[x/c]C}\ \text{ALL-RIGHT}}{\Gamma \ \vdash \ \forall x.\phi', \forall x.\phi', \Delta \ \Downarrow \forall x.\forall y.C}\ \text{ALL-RIGHT}$$

We can transform the subproof of $\Gamma \ \vdash \ [x/c]\phi', [x/d]\phi', \Delta \ \Downarrow [y/d][x/c]C$ into a proof of $\Gamma \ \vdash \ [x/c]\phi', [x/c]\phi', \Delta \ \Downarrow [y/c][x/c]C$ by replacing d everywhere in the proof with c. Subsequently, the subproof can be transformed into a proof of $\Gamma \ \vdash \ [x/c]\phi', \Delta \ \Downarrow D$ with $[y/c][x/c]C \Rightarrow D$ by the induction hypothesis. Simultaneously, one of the formulae $\forall x.\phi'$ and one of the ALL-RIGHT applications can be eliminated. Finally, it can be observed that the implication $\forall x.\forall y.C \Rightarrow \forall x.[y/x]C \Rightarrow \forall c.D$ holds.

– ϕ starts with $\forall$ and is in the antecedent, or with $\exists$ and is in the succedent, or is an equation, an inequality, a divisibility judgement or an atom $p(\bar{t})$. Because such formulae are not eliminated by any rule application, the two occurrences of ϕ can directly be replaced with only one occurrence everywhere in the proof.

Proof (Lem. 8). Suppose $\bar{a} = (a_1, \ldots, a_n)$ are the quantified variables, $c_1, \ldots, c_n$ are fresh constants and $\sigma = [a_1/c_1, \ldots, a_n/c_n]$. Let $\sigma_1, \ldots, \sigma_m$ be substitutions of $c_1, \ldots, c_n$ with integer literals that describe all solutions of $\sigma(\phi)$. By Lem. 7, there are $\mathrm{PresPred}^C_S$-proofs of the sequents $(\sigma_i(\sigma(\psi)) \ \vdash \quad \Downarrow C_i)_{i=1..m}$ for appropriate valid constraints $C_1, \ldots, C_m$. By Lem. 23, this implies that there are also m proofs of $(\sigma(\psi) \ \vdash \quad \Downarrow D_i)_{i=1..m}$ such that $\sigma_i(D_i)$ is valid for each i.

Then there is also a single $\mathrm{PresPred}^C_S$-proof $\sigma(\psi) \ \vdash \quad \Downarrow D$ such that $\sigma_i(D)$ is valid for each i: with the help of Lem. 24, we can normalise each $\mathrm{PresPred}^C_S$-proof $(\sigma(\psi) \ \vdash \quad \Downarrow D_i)_{i=1..m}$ to a proof where the first steps are applications of

AND-*, OR-*, NOT-*:

$$\dfrac{\Gamma_1 \vdash \Delta_1 \Downarrow D_{i,1} \quad \cdots \quad \Gamma_k \vdash \Delta_k \Downarrow D_{i,k}}{\begin{array}{c}\vdots\\ \sigma(\psi) \vdash \quad \Downarrow D_i\end{array}}$$

such that all formulae in $\Gamma_1, \Delta_1, \ldots, \Gamma_k, \Delta_k$ are atoms or formulae that start with a universal quantifier. This means that we can assume that each of the proofs $(\sigma(\psi) \vdash \quad \Downarrow D_i)_{i=1..m}$ contains a subproof for each sequent $(\Gamma_j \vdash \Delta_j)_{j=1..k}$, and that $D_i = D_{i,1} \wedge \cdots \wedge D_{i,k}$. These subproofs can be put together to create one general proof for each of $(\Gamma_j \vdash \Delta_j)_{j=1..k}$, because every (closed) goal of a PresPred_S^C-proof of $\Gamma_j \vdash \Delta_j$ contains $\Gamma_j \vdash \Delta_j$ as a sub-sequent (put the proofs together by starting with one proof, copy the second proof to all goals of the first proof, etc). When CLOSE is applied such that as many formulae as possible are selected in every goal, then for the constraint of the big proof for $\Gamma_j \vdash \Delta_j \Downarrow E_j$ the implication $D_{1,j} \vee \cdots \vee D_{m,j} \Rightarrow E_j$ holds. Finally, the big proofs can be assembled further to obtain the anticipated proof of $\sigma(\psi) \vdash \quad \Downarrow D$:

$$\dfrac{\begin{array}{c}\vdots\\ \Gamma_1 \vdash \Delta_1 \Downarrow E_1\end{array} \quad \cdots \quad \begin{array}{c}\vdots\\ \Gamma_k \vdash \Delta_k \Downarrow E_k\end{array}}{\begin{array}{c}\vdots\\ \sigma(\psi) \vdash \quad \Downarrow E_1 \wedge \cdots \wedge E_k\end{array}}$$

Because $\sigma_i(D_i)$ is valid for each i and $D_i = D_{i,1} \wedge \cdots \wedge D_{i,k}$, for each i the formula $\sigma_i(D) = \sigma_i(E_1 \wedge \cdots \wedge E_k)$ is valid as well.

Finally, we can prove $\exists \bar{a}.(\phi \wedge \psi)$ as follows:

$$\dfrac{\dfrac{\begin{array}{c}*\\ \vdots\\ \sigma(\phi), \sigma(\psi) \vdash \quad \Downarrow \neg\sigma(\phi) \vee D\end{array}}{\sigma(\phi \wedge \psi) \vdash \quad \Downarrow \neg\sigma(\phi) \vee D}\ \text{OR-LEFT}}{\exists \bar{a}.(\phi \wedge \psi) \vdash \quad \Downarrow \exists \bar{a}.(\neg\phi \vee [c_1/a_1, \ldots, c_n/a_n]D)}\ \text{EX-LEFT}^*$$

This proves the lemma, because $\exists \bar{a}.(\neg\phi \vee [c_1/a_1, \ldots, c_n/a_n]D)$ is a valid formula in Presburger Arithmetic.

Lemma 10 (Soundness of PresPred^C)

It is enough to check that all rules of the calculus are sound, which is trivial for most of the rules (also see Lem. 2). The interesting case is the rule OMEGA-ELIM. Assume that the lower sequent does not hold, i.e., C holds and the sequent

$$\Gamma, \{\alpha_i c - a_i \dot{\geq} 0\}_i, \{\beta_j c - b_j \dot{\leq} 0\}_j \vdash \Delta$$

is violated. This implies that the inequalities $\{\alpha_i c - a_i \dot{\geq} 0\}_i$, $\{\beta_j c - b_j \dot{\leq} 0\}_j$ hold. From the proof for Thm. 9 that is given in [10] we can conclude that then either the dark shadow conjunction $\bigwedge_{i,j} \alpha_i b_j - a_i \beta_j - (\alpha_i - 1)(\beta_j - 1) \dot{\geq} 0$ holds, or otherwise one of the splinters $\alpha_i c - a_i - k \doteq 0 \ \wedge \bigwedge_i \alpha_i c - a_i \dot{\geq} 0 \wedge \bigwedge_j \beta_j c - b_j \dot{\leq} 0$ has to be satisfied (note, that this is more than what is guaranteed by the actual Thm. 9, where the splinters are existentially quantified).

Lemma 11 (Constraint completeness in exhaustive proofs)

The conditions on page 177 ensure that (1) is preserved by all rule applications: if (1) holds for all premisses of a rule application, then it also holds for the conclusion. With the help of a simple induction, this entails that (1) holds for all proof nodes. In detail:

1. It is easy to see that AND-*, OR-*, NOT-*, PRED-UNIFY, RED, DIV-LEFT, DIV-RIGHT, and SIMP preserve (1). Observe that if $U' \subseteq U$, then:

$$\forall U'.\ (\mathit{\Gamma}_p \ \vdash\ \mathit{\Delta}_p) \ \Rightarrow\ \forall U'.\ C \quad \text{entails} \quad \forall U.\ (\mathit{\Gamma}_p \ \vdash\ \mathit{\Delta}_p) \ \Rightarrow\ \forall U.\ C$$

2. We only show the proof for EX-RIGHT-D. Let U be the annotation of the conclusion, ϕ a PA formula and assume $(\mathit{\Gamma}_p \ \vdash\ [x/c]\phi, \mathit{\Delta}_p) \ \Rightarrow\ [x/c]C$. This implies:

$$\forall U.\ (\mathit{\Gamma}_p \ \vdash\ \exists x.\phi, \mathit{\Delta}_p) \ \Leftrightarrow\ \forall U.\ \exists x.\ (\mathit{\Gamma}_p \ \vdash\ \phi, \mathit{\Delta}_p) \ \Rightarrow\ \forall U.\ \exists x.C$$

3. We show the proof for ALL-RIGHT. If ϕ is a PA formula, then:

$$\begin{aligned} &\forall U.\ (\mathit{\Gamma}_p \ \vdash\ \forall x.\phi, \mathit{\Delta}_p) \ \Leftrightarrow\ \forall (U \cup \{c\}).\ (\mathit{\Gamma}_p \ \vdash\ [x/c]\phi, \mathit{\Delta}_p) \\ &\quad \Rightarrow\ \forall (U \cup \{c\}).\ [x/c]C \ \Leftrightarrow\ \forall U.\ \forall x.C \end{aligned}$$

   Similarly, if ϕ contains uninterpreted predicates:

$$\begin{aligned} &\forall U.\ (\mathit{\Gamma}_p \ \vdash\ \mathit{\Delta}_p) \ \Leftrightarrow\ \forall (U \cup \{c\}).\ (\mathit{\Gamma}_p \ \vdash\ \mathit{\Delta}_p) \\ &\quad \Rightarrow\ \forall (U \cup \{c\}).\ [x/c]C \ \Leftrightarrow\ \forall U.\ \forall x.C \end{aligned}$$

4. For COL-RED:

$$\begin{aligned} &\forall U.\ \big(\mathit{\Gamma}_p, \alpha c + t \doteq 0 \ \vdash\ \mathit{\Delta}_p\big) \\ &\quad \Leftrightarrow\ \ \forall U.\ \big(\mathit{\Gamma}_p, \exists c'.(\alpha(u + c') + t \doteq 0 \wedge c - u - c' \doteq 0) \ \vdash\ \mathit{\Delta}_p\big) \\ &\quad \Leftrightarrow\ \ \forall (U \cup \{c'\}).\ \big(\mathit{\Gamma}_p, \alpha(u + c') + t \doteq 0, c - u - c' \doteq 0 \ \vdash\ \mathit{\Delta}_p\big) \\ &\quad \Rightarrow\ \ \forall (U \cup \{c'\}).\ [x/c']C \\ &\quad \Leftrightarrow\ \ \forall U.\ \forall x.C \end{aligned}$$

5. Let U be the annotation of the conclusion and assume:

$$\forall U.\ (\mathit{\Gamma}_p, \alpha(u + c') + t \doteq 0, c - u - c' \doteq 0 \ \vdash\ \mathit{\Delta}_p) \ \Rightarrow\ \forall U.\ [x/c']C$$

Because $c' \notin U$ and $c - u$ does not contain any constants from U, we can substitute $c - u$ for c':

$$\forall U.\ (\mathit{\Gamma}_p, \alpha(u + (c - u)) + t \doteq 0, c - u - (c - u) \doteq 0 \ \vdash\ \mathit{\Delta}_p) \quad\Rightarrow \quad \forall U.\ [x/c - u]C$$

which entails:

$$\forall U.\ (\mathit{\Gamma}_p, \alpha c + t \doteq 0 \ \vdash\ \mathit{\Delta}_p) \ \Rightarrow\ \forall U.\ [x/c - u]C$$

6. Follows directly from Thm. 9 and the fact that $c \in U$ does not occur in $\mathit{\Gamma}$ and $\mathit{\Delta}$.
7. Let U be the annotation of the conclusion ($c \in U$) and assume:

$$\forall U.\ (\mathit{\Gamma}_p, \alpha c - t \doteq 0 \ \vdash\ \mathit{\Delta}_p) \ \Rightarrow\ \forall U.\ C,$$
$$\forall U.\ (\mathit{\Gamma}_p, \alpha c - t \doteq 0 \ \vdash\ \mathit{\Delta}_p)$$

which directly entails (because $c \in U$ does not occur in C' and $C \Leftrightarrow [x/\alpha c]C'$ holds):

$$\forall U.\ [x/\alpha c]C' \ \Leftrightarrow\ \forall U.\ \forall x.\ (C' \vee \alpha \nmid x)$$

The last formula also holds if t is substituted for x:

$$\forall U.\ ([x/t]C' \vee \alpha \nmid t)$$

8. Assume $\mathit{val}_{I,\delta}(\forall U.\ \neg\phi_1 \vee \cdots \vee \neg\phi_n \vee \psi_1 \vee \cdots \vee \psi_m) = \mathit{ff}$ for some interpretation I and constant assignment δ. Let $U_2 \subseteq U$ be those U-constants that only occur in equations of the succedent $\psi_1, \ldots, \psi_m, \mathit{\Delta}$. We then modify δ appropriately, resulting in the assignment δ':
   - $\delta|_{U_2}$ is chosen such that all U_2-equations of the succedent evaluate to ff. This is possible because U_2-equations describe hyperplanes in $\mathbb{Z}^{|U_2|}$, and the intersection of the complements of (finitely many) hyperplanes is non-empty.

   Then: $\mathit{val}_{I,\delta'}(\mathit{\Gamma}_p, \phi_1, \ldots, \phi_n \ \vdash\ \psi_1, \ldots, \psi_m, \mathit{\Delta}_p) = \mathit{ff}$: formulae that do not contain U_2-constants evaluate to tt (antecedent) or ff (succedent) by assumption, and equations with U_2-constants by construction.

Lemma 14 (Termination and exhaustiveness)

Termination has to be proven on different levels:

Loop 1–2. Terminates because $<_r$ is well-founded.

As a special case, note that if any rule application derives the formula *false* (which is abbreviation for $1 \doteq 0$) in the antecedent, subsequent applications of RED will immediately replace *all* terms in the sequent with 0 and thereby cause the algorithm to terminate.

Loop 1–4. We prove termination similarly as in the appendix of [8], by considering the following mapping of sequents to triples $(c_U, c_E, |U|)$ of two multisets over $\mathbb{N} \cup \{\infty\}$ and one natural number. We call a constant c *dependent* if it occurs as the leading term of an equation $c + t \doteq 0$ in the antecedent, and *independent* otherwise.

- c_U is the multiset of greatest common divisors of leading coefficients for independent U-constants:
$$\left\{\!\!\left\{ \gcd(\alpha_1, \ldots, \alpha_n) \in \mathbb{N} \cup \{\infty\} \;\middle|\; \begin{array}{c} c \in U \text{ an independent constant,} \\ \alpha_1 c + t_1 \doteq 0, \ldots, \alpha_n c + t_n \doteq 0 \\ \text{all equations in the antecedent} \\ \text{whose leading term is } c \end{array} \right\}\!\!\right\}$$
in which we define $\gcd() = \infty$.
- c_E is the corresponding multiset for non-U-constants:
$$\left\{\!\!\left\{ \gcd(\alpha_1, \ldots, \alpha_n) \in \mathbb{N} \cup \{\infty\} \;\middle|\; \begin{array}{c} c \notin U \text{ an independent constant} \\ \text{in the sequent,} \\ \alpha_1 c + t_1 \doteq 0, \ldots, \alpha_n c + t_n \doteq 0 \\ \text{all equations in the antecedent} \\ \text{whose leading term is } c \end{array} \right\}\!\!\right\}$$
- $|U|$ is the number of U-constants.

Such triples $(c_U, c_E, |U|)$ are compared lexicographically. Multisets over $\mathbb{N} \cup \{\infty\}$ are compared using the well-founded ordering $<_m$: for elements $a_1 \leq \cdots \leq a_n$, $b_1 \leq \cdots \leq b_m$, we define:

$$\begin{aligned} &\{\{a_1, \ldots, a_n\}\} <_m \{\{b_1, \ldots, b_m\}\} \quad \text{iff} \\ &\qquad\qquad n < m \text{ or } (n = m \text{ and } (a_1, \ldots, a_n) <_{\text{lex}} (b_1, \ldots, b_m)) \end{aligned}$$

We prove termination of the loop 1–4 by showing that the triple $(c_U, c_E, |U|)$ for a sequent becomes strictly smaller each time step 3 or 4 is carried out, and does not become bigger if step 2 occurs. We only consider sequents in which the rule SIMP has been fully applied to all formulae (in other words, trailing applications of SIMP are conceptually considered as part of the other rule applications).

- RED (step 2): U does not change in this step.
  If the target formula is not an equation in the antecedent, the only relevant effect might be that constants disappear from a sequent, which does not increase the measure.
  Thus, assume that the application turns the left-hand side of an antecedent equation $s \doteq 0$ into $s + \alpha \cdot t <_r s$; after a possibly following application of SIMP, the new equation is $s' \doteq 0$:
  - if $s' = 0$, then it must be the case that $s = t$, which contradicts the assumption that $\phi[s]$ is not an equation in the antecedent.
  - if $s' = 1$, *false* has been derived and the strategy terminates abruptly.

  - if s and s' have the same leading term c and leading coefficients β, β', then $\beta' = \beta$ or $\beta' <_r \beta$. This implies that neither c_U nor c_E have become $<_m$-bigger.
  - if s and s' have different leading terms c, c' and leading coefficients β, β', then $c' <_r c$ and $t \doteq 0$ has the form $\gamma c + w \doteq 0$ where β is a multiple of γ.

    This implies that the c_U- or c_E-element for c has not changed (if c is dependent, there is no element at all). The c_U- or c_E-element for c' has at least not become bigger (again, it is possible that c' is dependent and there is not element).

- COL-RED-SUBST (step 3): first, note that t contains constants (otherwise, SIMP would be applicable), but does not contain U-constants (because $c \notin U$ and U-constants are $<_r$-maximal). Because no U-constants are involved, c_U stays the same. Further, the leading term of the equation $\alpha(u + c') + t \doteq 0$, after a potential application of SIMP, is not c': this could only be the case if all coefficients in t were multiples of α, which means that SIMP would have been applicable to $\alpha c + t \doteq 0$.

  If the leading coefficient of the new equation $\alpha(u + c') + t \doteq 0$ is 1, then the cardinality of c_E decreases (because an independent constant disappears) and c_E becomes $<_m$-smaller.

  Otherwise, there are three changes affecting c_E:
  - The constant c is independent before the rule application and dependent afterwards, which means that one element of c_E disappears. Because RED has been applied exhaustively before step 3, $\alpha c + t \doteq 0$ is the only equation in the antecedent whose leading term is c and the removed element is α.
  - The new constant c' is independent and does not occur as leading term of any equation, which means that ∞ is added as a new element to c_E.
  - The c_E-element belonging to the leading term d of the new equation $\alpha(u + c') + t \doteq 0$ (which was an independent constant before applying COL-RED-SUBST because RED was applied exhaustively) changes: suppose γ is the c_E-element belonging to d before the application of COL-RED-SUBST. Because u is chosen such that:

    $$(\alpha u + t) \;=\; \min_{<_r} \{\alpha u' + t \mid u' \ \text{ a term}\}$$

    the leading coefficient γ' of $\alpha(u + c') + t \doteq 0$, after a potential application of SIMP, has to be greater than 1 but strictly smaller than α. Besides, γ' is also strictly smaller than γ, because RED was applied exhaustively: if $\gamma < \infty$, there has to be an equation $\gamma d + w \doteq 0$ in the antecedent that can be applied to reduce t.

  Altogether, the new value of c_E is:

  $$c'_E \;=\; c_E \backslash \{\{\alpha, \gamma\}\} \cup \{\{\infty, \gamma'\}\}$$

  and $c'_E <_m c_E$ because of $0 < \gamma' < \alpha$ and $0 < \gamma' < \gamma$.

- COL-RED (step 3): shown in the same way as for COL-RED-SUBST, with the difference that c_U is considered instead of c_E. The condition that t contains further U-constants whose coefficient is not a multiple of α is needed to ensure that c' is not the leading term of the new equation $\alpha(u + c') + t \doteq 0$.
- DIV-CLOSE (step 4):
  - If $\alpha > 1$, then c was a independent U-constant before applying DIV-CLOSE, i.e., the cardinality of c_U becomes smaller because c is removed from U.
  - If $\alpha = 1$ and t contains further constants, then after an application of SIMP to the equation $\alpha c + t \doteq 0$ the constant c is no longer the leading term. Call the new leading term c' and its coefficient α'. The constant c' was independent before applying DIV-CLOSE. Because RED was applied exhaustively, the old c_U- or c_E-element for c' is bigger than α'. This implies that either c_U becomes $<_m$-smaller, or c_U stays the same and c_E becomes $<_m$-smaller.
  - If $\alpha = 1$ and t does not contain further constants, then neither c_U nor c_E changes, but the cardinality of U decreases.

Loop 1–5. It can first be observed that there is little interaction between the rule FM-ELIM and the rules of step 1–4 that treat equalities: step 5 is only reached once the leading coefficient of all equations in the antecedent is 1, and once no leading term of such an equation occurs in more than place in the sequent. This implies that an application of FM-ELIM never enables further applications of the rules in step 1–4. The application of FM-ELIM alone has to terminate because for any finite set of inequalities there is only a finite number of Fourier-Motzkin-inferences (respecting the ordering $<_r$).

In order to show that the application of ANTI-SYMM, AND-*, OR-*, NOT-* terminates, we use a pair of natural numbers as a measure for the complexity of a sequent. Two such pairs are compared lexicographically:

- The number of propositional connectives $\wedge$, $\vee$, $\neg$ in the sequent.
- The dimension d of the smallest affine space in $\mathbb{R}^C$ that contains all (integer) solutions of the equations in the antecedent, where C is the set of all constants occurring in a sequent.

Both FM-ELIM and the rules in step 1–4 do not apply to propositional connectives and preserve the dimension d, while the other rules of step 5 decrease the measure:

- ANTI-SYMM (step 5): again, this rule is only applied once the leading coefficient of all equations in the antecedent is 1, and once no leading term of such an equation occurs in more than place in the sequent (in particular not in the equation that is added by ANTI-SYMM). This implies that the dimension d is decreased by 1 by the new equation.
- AND-*, OR-*, NOT-* (step 5): eliminates one propositional connective.

Loop 1–6. Again, we use vectors of natural numbers that are compared lexicographically as a measure for sequents:

- The number of divisibility judgements $\alpha \mid t$ in positive positions.
- The number of divisibility judgements $\alpha \mid t$ in negative positions.
- The number of quantifiers $\forall$, $\exists$ in the sequent.
- The dimension d of the smallest affine space in $\mathbb{R}^C$ that contains all (integer) solutions of the equations in the antecedent, where C is the set of all constants occurring in a sequent.
- The number of equations in positive positions.

None of the rules in step 1–5 increase any of these features (but possibly decrease some of them), while the other rules of step 6 decrease the measure:

- SPLIT-EQ (step 6): eliminates an equation in a positive position.
- OMEGA-ELIM (step 6): an application of this rule will at first not have any influence on the complexity of a sequent (but it introduces new propositional connectives). The next rules applicable after OMEGA-ELIM, however, are the rules AND-*, OR-*, NOT-* of step 5 that split the introduced formula into its disjuncts. For each of the disjuncts, the dimension d is reduced by 1: the first of the disjuncts does no longer contain the constant c at all (the set C becomes smaller), while the other disjuncts introduce a new equation in a negative position so that the same argument as for the rule ANTI-SYMM applies.
- ALL-*, EX-* (step 6): eliminates one quantifier.
- DIV-RIGHT (step 6): eliminates a divisibility judgement in a positive position (and introduces new judgements in negative positions).
- DIV-LEFT (step 6): eliminates a divisibility judgement in a negative position.

Exhaustiveness. To prove that the resulting proof is exhaustive, annotate the proof tree with the sets U that are maintained by the strategy. The most involved point is to see that CLOSE is applied in the right way. To this end, observe that if CLOSE must not be applied according to the conditions in Sect. 5.1, then some other rule with higher priority can be applied.

Lemma 15 (Quantifier elimination)

By induction on the size of a proof, show that if a sequent $\Gamma \vdash \Delta \Downarrow C$ of the resulting exhaustive proof is annotated with U, then C does not contain any U-constants. Note, in particular, that divisibility judgements $\alpha \mid t$ (which can equivalently be expressed using existentially quantified equations) never reduce the set U. Because the introduced constant is added to U when the rules ALL-RIGHT, EX-LEFT or COL-RED are applied, this proves the lemma.

Lemma 16 ($\mathbf{PresPred}_S^C$ subsumes $\mathbf{PresPred}^C$)

We first need two further lemmas:

Lemma 25. *Suppose a $\mathit{PresPred}_S^C$-proof exists for the sequent $\Gamma \vdash \Delta \Downarrow C$. For some D with $C \Rightarrow D$ there is a proof of the sequent $\Gamma \vdash \Delta \Downarrow D$ in which no rule apart from* CLOSE *is applied to formulae that do not contain uninterpreted predicates (i.e., to PA formulae).*

Proof. The prove is done by induction on the size of the original proof P of $\Gamma \vdash \Delta \Downarrow C$. We can assume that the first rule application in P is performed on a PA formula ϕ, and that no other rule application in P (apart from CLOSE) involves PA formulae. Furthermore, by Lem. 24, we can assume that uninterpreted predicates occur in $\Gamma \vdash \Delta$ only in formulae in the antecedent that start with $\forall$, in formulae in the succedent that start with $\exists$, or in literals $p(\bar{t})$ (otherwise, by Lem. 24 we can turn P into a proof in which the first rule application is performed on a formula with unint. predicates of different shape and consider the direct subproofs of this proof). There are the following cases:

- ϕ starts with $\neg$, with $\wedge$ and is in the antecedent, or with $\vee$ and is in the succedent. Because no rules apart from CLOSE are applied to the formulae resulting from the first rule application in P, the application can simply be left out without changing the overall constraint.
- ϕ starts with $\vee$ and is in the antecedent, or with $\wedge$ and is in the succedent, i.e., the first rule application splits P into two subproofs. E.g.:

$$\dfrac{\begin{array}{c}\vdots\\ \Gamma \vdash \phi_1, \Delta \Downarrow C\end{array} \qquad \begin{array}{c}\vdots\\ \Gamma \vdash \phi_2, \Delta \Downarrow D\end{array}}{\Gamma \vdash \phi_1 \wedge \phi_2, \Delta \Downarrow C \wedge D} \text{ AND-RIGHT}$$

  Because no further rules (apart from CLOSE) are applied to ϕ_1 and ϕ_2 (or to any PA formulae), this means that there are proofs of $\Gamma \vdash \Delta \Downarrow C'$ and $\Gamma \vdash \Delta \Downarrow D'$ such that $C \Rightarrow C' \vee \phi_1$ and $D \Rightarrow D' \vee \phi_2$. Further, because of the assumption about the formulae in Γ, Δ, we know that $\Gamma \vdash \Delta$ is a subsequent of each goal in both subproofs. This means that we can copy the second subproof to each goal of the first subproof (possibly renaming constants so that no name clashes occur). If CLOSE is in each goal applied as liberally as possible, the constraint of the resulting proof is at least as weak as $C' \vee D'$. Finally, by adding $\phi_1 \wedge \phi_2$ to all succedents in the proof, the constraint can be made as weak as $C' \vee D' \vee \phi_1 \wedge \phi_2$. Because of $C \wedge D \Rightarrow (C' \vee \phi_1) \wedge (D' \vee \phi_2) \Rightarrow C' \vee D' \vee \phi_1 \wedge \phi_2$, this concludes the case.
- ϕ starts with $\exists$ and is in the antecedent, or with $\forall$ and is in the succedent. E.g.:

$$\dfrac{\begin{array}{c}\vdots\\ \Gamma \vdash [x/c]\phi', \Delta \Downarrow [x/c]C\end{array}}{\Gamma \vdash \forall x.\phi', \Delta \Downarrow \forall x.C} \text{ ALL-RIGHT}$$

As before, this means that there is a proof of a sequent $\Gamma \vdash \Delta \Downarrow D$ such that $[x/c]C \Rightarrow D \vee [x/c]\phi'$, whereby we can assume that c does not occur in D. By adding $\forall x.\phi'$ to all succedents of this proof, we obtain a proof of $\Gamma \vdash \forall x.\phi', \Delta \Downarrow E$ such that $D \vee \forall x.\phi' \Rightarrow E$. Altogether, this means that $\forall x.C \Rightarrow \forall x.(D \vee \phi') \Rightarrow D \vee \forall x.\phi' \Rightarrow E$.

- ϕ starts with $\forall$ and is in the antecedent, or with $\exists$ and is in the succedent. E.g.

$$\dfrac{\begin{array}{c}\vdots\\ \Gamma \vdash [x/c]\phi', \exists x.\phi', \Delta \Downarrow [x/c]C\end{array}}{\Gamma \vdash \exists x.\phi', \Delta \Downarrow \exists x.C}\ \text{EX-RIGHT}$$

  By leaving out $[x/c]\phi'$ everywhere, we obtain a proof of $\Gamma \vdash \exists x.\phi', \Delta \Downarrow D$ such that $[x/c]C \Rightarrow D \vee [x/c]\phi'$, whereby we can assume that c does not occur in D. If CLOSE is applied as liberally as possible in each goal, the implication $\exists x.\phi' \Rightarrow D$ holds, i.e., $D \Leftrightarrow D \vee \exists x.\phi'$. Altogether, this means $\exists x.C \Rightarrow \exists x.(D \vee \phi') \Rightarrow D \vee \exists x.\phi' \Rightarrow D$.

The following lemma will be used to justify application of the rules RED and SIMP:

Lemma 26. *Suppose that Γ_p, Δ_p are sets of PA formulae and s_1, s_2 are two terms or two PA atoms (equations, inequalities, or divisibility judgements) such that:*

$$\begin{array}{lcll}\bigwedge \Gamma_p \to \bigvee \Delta_p & \Rightarrow & s_1 - s_2 \doteq 0 & \textit{(in case of terms)}\\ \bigwedge \Gamma_p \to \bigvee \Delta_p & \Rightarrow & (s_1 \wedge s_2) \vee (\neg s_1 \wedge \neg s_2) & \textit{(in case of atoms)}\end{array}$$

Further, suppose a $\mathit{PresPred}^C_S$-proof exists for $\Gamma, \Gamma_p \vdash \phi[s_1], \Delta_p, \Delta \Downarrow C$ (we write $\phi[s_1]$ in the succedent to denote that the term or atom s_1 can occur in an arbitrary position in the sequent, in particular also in the antecedent) in which no rule apart from CLOSE is applied to formulae that do not contain uninterpreted predicates. For some D with $C \Rightarrow D$ there is a proof of $\Gamma, \Gamma_p \vdash \phi[s_2], \Delta_p, \Delta \Downarrow D$ that has the same depth as the original proof and that starts with the same rule application, and in which no rule apart from CLOSE is applied to formulae that do not contain uninterpreted predicates.

Proof. The prove is done by induction on the size of the original proof P of $\Gamma, \Gamma_p \vdash \phi[s_1], \Delta_p, \Delta \Downarrow C$. As in the proof of Lem. 24, we first show the main conjecture using proof trees with multiset sequents, and then refer to *Step 2* of the proof of Lem. 24 to carry over the result to proofs with normal sequents. Observe that none of the following transformation steps increases the depth of a proof or introduces new rule applications (other than CLOSE) to PA formulae.

We can assume that the first rule application in P involves the formula $\phi[s_1]$. Otherwise, consider the maximal subproofs of P with this property. Outside of the subproofs, $\phi[s_1]$ can simply be replaced with $\phi[s_2]$ and is unaffected by other rule applications due to the usage of multiset sequents.

If the first (and only) rule application in P is CLOSE and includes $\phi[s_1]$, the formula cannot contain uninterpreted predicates, and then neither does $\phi[s_2]$. This means that $\phi[s_1]$ can be replaced with $\phi[s_2]$ in this goal. CLOSE can be applied such that both $\phi[s_2]$ and the formulae Γ_p, Δ_p are included. Because of $\phi[s_1] \Rightarrow (\bigwedge \Gamma_p \rightarrow \bigvee \Delta_p) \rightarrow \phi[s_2]$, the resulting constraint does not get stronger.

If the first application in P is done with a rule other than CLOSE, it is always possible to do this application to $\phi[s_2]$ instead of $\phi[s_1]$ and to handle the direct subproofs using the induction hypothesis (note, that $\phi[s_1]$ and $\phi[s_2]$ have the same structure).

Proof (Lem. 16). By an induction on the size of the PresPred^C-proof P. In each step, it can be assume that the first rule application in P is done using a rule that is not present in PresPred^C_S, and that all other rules applied in P are PresPred^C_S-rules. By Lem. 25, we can furthermore assume that no inner rule application (other than CLOSE) is done on PA formulae. The following cases are possible, depending on the first rule applied:

- ALL-LEFT-D, EX-RIGHT-D: replace the application with ALL-LEFT or EX-RIGHT, which does not make the resulting constraint stronger.
- RED, SIMP: the rule application can be left out with the help of Lem. 26.
- DIV-LEFT, DIV-RIGHT, ANTI-SYMM, FM-ELIM: because these rules replace PA formulae with equivalent formulae, they can directly be left out without strengthening constraints.
- COL-RED, e.g.:

$$\dfrac{\begin{array}{c}\vdots\\ \Gamma, \alpha(u + c') + t \doteq 0, c - u - c' \doteq 0 \;\vdash\; \Delta \;\Downarrow [x/c']C'\end{array}}{\Gamma, \alpha c + t \doteq 0 \;\vdash\; \Delta \;\Downarrow \forall x. C'}\;\text{COL-RED}$$

By leaving out the two equations in the antecedent, we can create a proof of a sequent $\Gamma \vdash \Delta \Downarrow C''$ with:

$$[x/c']C' \quad\Rightarrow\quad C'' \vee \alpha(u + c') + t \not\doteq 0 \vee c - u - c' \not\doteq 0$$

whereby it can be assumed that c' does not occur in C''. Then, by adding $\alpha c + t \doteq 0$ to all antecedents, a proof of $\Gamma, \alpha c + t \doteq 0 \vdash \Delta \Downarrow D$ can be derived such that $C'' \vee \alpha c + t \not\doteq 0 \Rightarrow D$. Altogether, this means:

$$\begin{aligned}\forall x. C' \quad\Rightarrow\quad \forall c'.[x/c']C' \quad\Rightarrow\quad & \forall c'.(C'' \vee \alpha(u + c') + t \not\doteq 0 \vee c - u - c' \not\doteq 0)\\ \Rightarrow\quad & C'' \vee \alpha c + t \not\doteq 0 \quad\Rightarrow\quad D\end{aligned}$$

- COL-RED-SUBST: analogously.
- DIV-CLOSE, e.g.:

$$\dfrac{\begin{array}{c}\vdots\\ \Gamma, \alpha c - t \doteq 0 \;\vdash\; \Delta \;\Downarrow C'\end{array}}{\Gamma, \alpha c - t \doteq 0 \;\vdash\; \Delta \;\Downarrow [x/t]C'' \vee \alpha \nmid t}\;\text{DIV-CLOSE}$$

If CLOSE is in the proof always applied as liberally as possible, such that also the equation $\alpha c - t \doteq 0$ is selected, then $\alpha c - t \not\doteq 0 \Rightarrow C'$, i.e., the equivalence $C' \Leftrightarrow C' \vee \alpha c - t \not\doteq 0$ holds. Because of $C' \Leftrightarrow [x/\alpha c]C''$, this means $C' \Leftrightarrow [x/t]C'' \vee \alpha c - t \not\doteq 0$. Finally, because of $\alpha c - t \doteq 0 \Rightarrow \alpha \mid t$:

$$[x/t]C'' \vee \alpha \nmid t \quad \Rightarrow \quad [x/t]C'' \vee \alpha c - t \not\doteq 0 \quad \Rightarrow \quad C'$$

This means that the constraint of the proof does not become stronger if the application of DIV-CLOSE is left out.

- SPLIT-EQ, e.g.:

$$\dfrac{\begin{matrix}\vdots \\ \Gamma \vdash t \mathrel{\dot{\leq}} 0, \Delta \Downarrow C'\end{matrix} \qquad \begin{matrix}\vdots \\ \Gamma \vdash t \mathrel{\dot{\geq}} 0, \Delta \Downarrow D'\end{matrix}}{\Gamma \vdash t \doteq 0, \Delta \Downarrow C' \wedge D'}\ \textsc{split-eq}$$

We can modify the proof to get a similar one without the application of SPLIT-EQ:

$$\dfrac{\begin{matrix}\vdots \\ \Gamma \vdash t \mathrel{\dot{\leq}} 0, \Delta \Downarrow C'\end{matrix} \qquad \begin{matrix}\vdots \\ \Gamma \vdash t \mathrel{\dot{\geq}} 0, \Delta \Downarrow D'\end{matrix}}{\Gamma \vdash t \mathrel{\dot{\leq}} 0 \wedge t \mathrel{\dot{\geq}} 0, \Delta \Downarrow C' \wedge D'}\ \textsc{and-right}$$

By Lem. 25, this can be turned into a proof of $\Gamma \vdash t \mathrel{\dot{\leq}} 0 \wedge t \mathrel{\dot{\geq}} 0, \Delta \Downarrow E$ in which no rules other than CLOSE are applied to PA formulae, such that the implication $C' \wedge D' \Rightarrow E$ holds. Finally, $t \mathrel{\dot{\leq}} 0 \wedge t \mathrel{\dot{\geq}} 0$ can be replaced with the (equivalent) equation $t \doteq 0$ everywhere in the proof, which leads to a proof of $\Gamma \vdash t \doteq 0, \Delta \Downarrow E'$ with $E' \Leftrightarrow E$.

- OMEGA-ELIM, e.g.:

$$\dfrac{\begin{matrix}\vdots \\ \Gamma, \phi(c) \vdash \Delta \Downarrow C\end{matrix}}{\Gamma, \{\alpha_i c - a_i \mathrel{\dot{\geq}} 0\}_i, \{\beta_j c - b_j \mathrel{\dot{\leq}} 0\}_j \vdash \Delta \Downarrow C}\ \textsc{omega-elim}$$

The rule application can simply be left out because of:

$$\bigwedge_i \alpha_i c - a_i \mathrel{\dot{\geq}} 0 \wedge \bigwedge_j \beta_j c - b_j \mathrel{\dot{\leq}} 0 \quad \Rightarrow \quad \phi(c)$$

This implication follows from the proof for Thm. 9 that is given in [10] (note, that this is more than what is guaranteed by the actual Thm. 9, where the splinters are existentially quantified).

Lemma 17 (Fair proof construction)

We call the $\mathrm{PresPred}^C_S$-proof P and the fair $\mathrm{PresPred}^C$-proof Q. By Lem. 25, we can assume that the only rule that is applied to PA formulae in P is CLOSE.

For the following induction, we also make the assumption that the rule PRED-UNIFY is in Q applied in the same fair manner as the rules in Fig. 1, i.e., it is eventually applied infinitely often to all complementary predicate literals (or their successors). Given any fair PresPred^C-proof, it is possible to insert further applications of PRED-UNIFY to achieve this property without changing the constraints generated by the proof (the constraints stay equivalent). Namely, assume that PRED-UNIFY is applied at some point in the proof:

$$\dfrac{\begin{array}{c}\vdots\\ \Gamma, p(s_1,\ldots,s_n) \vdash p(t_1,\ldots,t_n), \bigwedge_i s_i - t_i \doteq 0, \Delta \Downarrow C\end{array}}{\Gamma, p(s_1,\ldots,s_n) \vdash p(t_1,\ldots,t_n), \Delta \Downarrow C}\ \text{PRED-UNIFY}$$

Let $\Gamma_p \subseteq \Gamma$ and $\Delta_p \subseteq \Delta \cup \{\bigwedge_i s_i - t_i \doteq 0\}$ be the sets of PA formulae in the premiss that do not contain existential quantifiers (no $\exists$ for formulae in the succedent, no $\forall$ for formulae in the antecedent). It is obviously the case that an immediate second application of PRED-UNIFY is unnecessary because of:

$$\bigwedge_i s_i - t_i \doteq 0 \quad\Rightarrow\quad \bigwedge \Gamma_p \rightarrow \bigvee \Delta_p \tag{3}$$

i.e., the conjunction introduced by a second application is subsumed by the formulae already present in the sequent. By a simple induction on the size of a PresPred^C-proof, it can be shown that property (3) is preserved when applying arbitrary PresPred^C-rules (including RED or SIMP to the complementary literals $p(s_1,\ldots,s_n)$, $p(t_1,\ldots,t_n)$).

We perform Noetherian induction on the set of all possible pairs (P, Q), where P is a PresPred^C_S-proof for the sequent $\Gamma \vdash \Delta \Downarrow C$ in which the only rule that is applied to PA formulae is CLOSE, and Q is a fair PresPred^C-proof of $\Gamma \vdash \Delta \Downarrow ?$ (fair also concerning PRED-UNIFY in the way described above). The ordering is the lexicographic order on the pair (d_P, n), where

- d_P is the length of the longest branch in P (the depth of P), and
- n is the maximum number of rule applications that happen on a branch of Q before the first rule application in P is done on the branch. Because of fairness, the first rule application in P is eventually performed on all Q-branches, although possibly on successors of the involved formulae. By König's lemma, the maximum number of other rule applications before this happens is finite. In case the first rule application in P is CLOSE, we define $n = 0$.

The induction hypothesis is:

> Suppose that the root of Q is annotated with U. Then Q generates a constraint D with $\forall U.C \Rightarrow \forall U.D$.

There are a number of induction step cases. In all of them, we assume that the constants introduced by EX-*, ALL-* are renamed when necessary to avoid collisions. Further, we make use of the fact that also all subproofs of Q are fair proofs (also concerning PRED-UNIFY).

- The first rule application in P is CLOSE. We can then simply prune Q and apply CLOSE to the same formulae as in P. In all of the following cases, it is therefore assumed that P does not start with CLOSE (which implies, because of fairness, that also Q does not start with CLOSE).
- P and Q start with the same rule application to the same formula(e). In case the rule is EX-* or ALL-*, we can ensure through renaming that the same constant is introduced. Then, we can apply the induction hypothesis to the direct subtrees of P and Q. There are the following cases, depending on the first rule applied:
  - AND-LEFT, OR-RIGHT, NOT-*, PRED-UNIFY: by the induction hypothesis, we know $\forall U'.C \Rightarrow \forall U'.D$ for the constraints C, D and annotation U' of the subtree roots. Because of $U' \subseteq U$, this entails $\forall U.\ C \Rightarrow \forall U.\ D$.
  - AND-RIGHT, OR-LEFT: by the induction hypothesis, $\forall U'.C' \Rightarrow \forall U'.D'$ and $\forall U''.C'' \Rightarrow \forall U''.D''$ for the constraints and annotations of the subtree roots. Because of $U' \subseteq U$ and $U'' \subseteq U$, this entails:

    $$\forall U.\ (C' \wedge C'') \quad \Rightarrow \quad \forall U.\ (D' \wedge D'')$$

  - ALL-LEFT, EX-RIGHT: by the induction hypothesis, $[x/c]C' \Rightarrow [x/c]D'$ for the constraints of the subtrees (which are annotated with the empty set), which entails $\forall U.\exists x.\ C' \Rightarrow \forall U.\exists x.\ D'$.
  - ALL-RIGHT, EX-LEFT: we know that $\forall U'.[x/c]C' \Rightarrow \forall U'.[x/c]D'$ for the constraints and annotations of the subtrees. Because of $U' \subseteq U \cup \{c\}$, this entails: $\forall U.\forall x.\ C' \Rightarrow \forall U.\forall x.\ D'$.

  In all of the following cases, we therefore assume that P and Q start with different rule applications.
- The first rule application in Q is AND-*, OR-*, NOT-*, EX-LEFT, ALL-RIGHT to a formula ϕ. By Lem. 24, we can transform P into a proof P' of some sequent $\Gamma \ \vdash\ \Delta \ \Downarrow D$ with $C \Rightarrow D$ that starts with the same rule application as Q. The depth of P' is at most one bigger than the depth of P, and the first rule application of P is the second rule application on all branches in P'. Furthermore, the only rule in P' that is applied to PA formulae is CLOSE, possibly apart from the first rule application in P'. We can then apply the induction hypothesis to the direct subtrees of P' and Q.
- The first rule application in Q is PRED-UNIFY, ALL-LEFT or EX-RIGHT. This rule application can be inserted as first rule application in P, adding the resulting formula to all sequents, which leads to a proof P' whose depth is one bigger than that of P and that has the same or a weaker constraint as P. The first rule application of P is the second rule application in P'. Then, the induction hypothesis can be applied to the direct subtrees of P' and Q.
- The first rule application in Q is EX-RIGHT-D or ALL-LEFT-D. E.g.:

$$\dfrac{\begin{array}{c}\vdots\\ \Gamma \ \vdash\ [x/c]\phi, \Delta \ \Downarrow ?\end{array}}{\Gamma \ \vdash\ \exists x.\phi, \Delta \ \Downarrow ?}\ \text{EX-RIGHT-D}$$

Because the proof P (of the sequent $\Gamma \vdash \exists x.\phi, \Delta \Downarrow C$) does not contain any rule applications to $\exists x.\phi$ apart from CLOSE (the formula does not contain uninterpreted predicates), this means that $\exists x.\phi$ can be left out everywhere in P, leading to a similar proof P' of a sequent $\Gamma \vdash \Delta \Downarrow C'$ with $C \Rightarrow C' \vee \exists x.\phi$ (as in the proof of Lem. 25). It is then possible to add the formula $[x/c]\phi$ to all succedents in P', resulting in a proof P'' of a sequent $\Gamma \vdash [x/c]\phi, \Delta \Downarrow C''$ (if necessary, one has to ensure by renaming that c does not occur in P'). If CLOSE is applied as liberally as possible in P'', the implication $C' \vee [x/c]\phi \Rightarrow C''$ holds. Finally, a proof P''' can be obtained from P'' by inserting EX-RIGHT-D as first rule application:

$$\dfrac{\begin{array}{c}\vdots\\ \Gamma \vdash [x/c]\phi, \Delta \Downarrow C''\end{array}}{\Gamma \vdash \exists x.\phi, \Delta \Downarrow \exists c.C''}\ \textsc{ex-right-d}$$

The depth of P''' is one bigger than the depth of P, and the first rule application in P is the second rule application in P'''. Thus, applying the induction hypothesis to the direct subproofs of P''' and Q, we know that $C'' \Rightarrow D'$ (the annotation of the root of the direct subproof of Q is the empty set). This entails that:

$$C \;\Rightarrow\; C' \vee \exists x.\phi \;\Rightarrow\; \exists x.(C' \vee \phi) \;\Rightarrow\; \exists c.(C' \vee [x/c]\phi) \;\Rightarrow\; \exists c.C'' \;\Rightarrow\; \exists c.D'$$

and therefore $\forall U.C \Rightarrow \forall U.\exists c.D'$.

- If the first rule application in Q is COL-RED, DIV-LEFT, DIV-RIGHT, SPLIT-EQ, ANTI-SYMM, or FM-ELIM, the same technique as in the previous case can be used.
- If the first rule application in Q is RED or SIMP, we can insert the same application as first step in P with the help of Lem. 26. Then, the induction hypothesis can be applied to the direct subproofs of the proofs.
- If the first rule application in Q is COL-RED-SUBST, we can first turn P into a proof P' of a sequent $\Gamma, \alpha(u + c') + t \doteq 0, c - u - c' \doteq 0 \vdash \Delta \Downarrow C'$ by replacing the original equation $\alpha c + t \doteq 0$ (if necessary, it has to be ensured by bound renaming that c' does not occur in P). If CLOSE is applied as liberally as possible in P', it holds that $C \Rightarrow C' \vee \alpha c + t \not\doteq 0$ and $\alpha(u + c') + t \not\doteq 0 \vee c - u - c' \not\doteq 0 \Rightarrow C'$. We can then obtain a proof P'' of $\Gamma, \alpha c + t \doteq 0 \vdash \Delta \Downarrow [c'/c - u]C'$ by adding COL-RED as first rule application in P'. Considering the constraints, we have:

$$\begin{aligned}&[c'/c - u](\alpha(u + c') + t \not\doteq 0 \vee c - u - c' \not\doteq 0)\\ &\qquad\qquad\qquad\qquad \Rightarrow\quad \alpha c + t \not\doteq 0 \quad\Rightarrow\quad [c'/c - u]C'\end{aligned}$$

Because C and u do not contain c', this altogether means that the implication $C \Rightarrow [c'/c - u](C' \vee \alpha c + t \not\doteq 0) \Rightarrow [c'/c - u]C'$ holds. Furthermore, applying the induction hypothesis to the direct subproofs of P'' and Q, we know that $\forall U'.C' \Rightarrow \forall U'.D'$ holds for the constraints and annotations of the subproofs. Because $c, c' \notin U'$ and u does not contain any constants from U, then also:

$$\forall U'.\ [c'/c - u]C' \quad\Rightarrow\quad \forall U'.\ [c'/c - u]D'$$

Finally, because of $U' \subseteq U$:

$$\forall U.C \quad\Rightarrow\quad \forall U.\ [c'/c - u]C' \quad\Rightarrow\quad \forall U.\ [c'/c - u]D'$$

- If the first rule application in Q is DIV-CLOSE, it is the case that $c \in U$ and we can simple insert DIV-CLOSE as first rule application in P, resulting in a proof P'. By the induction hypothesis, $\forall U'.C \Rightarrow \forall U'.D$ for the constraints and annotation of the direct subproofs, and because of $U' \subseteq U$ also $\forall U.C \Rightarrow \forall U.D$. Let D' be a formula with $D \Leftrightarrow [x/\alpha c]D'$ that does not contain c. Then:

$$\begin{aligned}\forall U.C \quad\Rightarrow\quad \forall U.D \quad\Rightarrow\quad \forall U.[x/\alpha c]D' \quad&\Rightarrow\quad \forall U.\forall x.(D' \vee \alpha \nmid x)\\ &\Rightarrow\quad \forall U.([x/t]D' \vee \alpha \nmid t)\end{aligned}$$

- The first rule application in Q is OMEGA-ELIM, which means that $c \in U$:

$$\frac{\begin{array}{c}\vdots\\ \Gamma, \phi(c) \ \vdash\ \Delta \Downarrow C\end{array}}{\Gamma, \{\alpha_i c - a_i \dot{\geq} 0\}_i, \{\beta_j c - b_j \dot{\leq} 0\}_j \ \vdash\ \Delta \Downarrow C}\ \textsc{omega-elim}$$

Because P does not contain any rule applications to the eliminated inequalities (other than CLOSE), these formulae can be left out everywhere, leading to a proof P' of the sequent $\Gamma \ \vdash\ \Delta \Downarrow C'$ with:

$$C \quad\Rightarrow\quad C' \vee \neg\Big(\bigwedge_i \alpha_i c - a_i \dot{\geq} 0 \wedge \bigwedge_j \beta_j c - b_j \dot{\leq} 0\Big)$$

Because Γ, Δ do not contain c, we can also assume that c does not occur in C'. Next, we can add the formula $\phi(c)$ to all antecedents, which yields a proof P'' of $\Gamma, \phi(c) \ \vdash\ \Delta \Downarrow C''$. If CLOSE is applied as liberally as possible in P'', the implication $C' \vee \neg\phi(c) \Rightarrow C''$ holds. Finally, OMEGA-ELIM can be inserted as first rule application in P'', which results in the proof P'''. The induction hypothesis can be applied to the direct subproofs of P''' and Q, which means that $\forall U'.C'' \Rightarrow \forall U'.D$ for the constraints and annotation of the subproofs. Because of $U' \subseteq U$, then also $\forall U.C'' \Rightarrow \forall U.D$. Furthermore:

$$\begin{aligned}\forall c.C \quad&\Rightarrow\quad \forall c.\Big(C' \vee \neg\Big(\bigwedge_i \alpha_i c - a_i \dot{\geq} 0 \wedge \bigwedge_j \beta_j c - b_j \dot{\leq} 0\Big)\Big)\\ &\Rightarrow\quad C' \vee \neg\exists c.\Big(\bigwedge_i \alpha_i c - a_i \dot{\geq} 0 \wedge \bigwedge_j \beta_j c - b_j \dot{\leq} 0\Big)\\ &\overset{(*)}{\Rightarrow}\quad C' \vee \neg\exists c.\phi(c)\\ &\Rightarrow\quad \forall c.(C' \vee \neg\phi(c))\\ &\Rightarrow\quad \forall c.C''\end{aligned}$$

where $(*)$ makes use of Thm. 9. Altogether, this entails $\forall U.C \Rightarrow \forall U.D$.

Lemma 19 (Shielded Constraints)

We first need a further lemma:

Lemma 27. *If x is a variable, $\phi_1, \ldots, \phi_m$ are formulae in which x does not occur, and $\psi_0[x], \ldots, \psi_m[x]$ are arbitrary formulae, then the following equivalence holds:*

$$\forall x.\Big(\psi_0[x] \vee \bigvee_{i=1}^{m} (\psi_i[x] \wedge \phi_i)\Big) \quad\Leftrightarrow\quad \bigvee_{S\subseteq\{1,\ldots,m\}} \Big(\bigwedge_{i\in S}\phi_i \wedge \forall x.\Big(\psi_0[x] \vee \bigvee_{i\in S}\psi_i[x]\Big)\Big)$$

Proof. By induction on m. The case $m = 0$ is clear, and the step case $m \to m+1$ as follows:

$$\begin{aligned}
& \forall x.\Big(\psi_0[x] \vee \bigvee_{i=1}^{m+1} (\psi_i[x] \wedge \phi_i)\Big) \\
\Leftrightarrow\ & \forall x.\Big((\psi_0[x] \vee \psi_{m+1}[x] \wedge \phi_{m+1}) \vee \bigvee_{i=1}^{m} (\psi_i[x] \wedge \phi_i)\Big) \\
\overset{\text{(IH)}}{\Leftrightarrow}\ & \bigvee_{S\subseteq\{1,\ldots,m\}} \Big(\bigwedge_{i\in S}\phi_i \wedge \forall x.\Big(\psi_0[x] \vee \psi_{m+1}[x] \wedge \phi_{m+1} \vee \bigvee_{i\in S}\psi_i[x]\Big)\Big) \\
\Leftrightarrow\ & \bigvee_{S\subseteq\{1,\ldots,m\}} \left(\bigwedge_{i\in S}\phi_i \wedge \left(\begin{array}{l} \forall x.(\psi_0[x] \vee \psi_{m+1}[x] \vee \bigvee_{i\in S}\psi_i[x]) \\ \wedge\ \forall x.(\psi_0[x] \vee \phi_{m+1} \vee \bigvee_{i\in S}\psi_i[x]) \end{array}\right)\right) \\
\overset{(*)}{\Leftrightarrow}\ & \bigvee_{S\subseteq\{1,\ldots,m\}} \left(\bigwedge_{i\in S}\phi_i \wedge \left(\begin{array}{l} \phi_{m+1} \wedge \forall x.(\psi_0[x] \vee \psi_{m+1}[x] \vee \bigvee_{i\in S}\psi_i[x]) \\ \vee\ \forall x.(\psi_0[x] \vee \bigvee_{i\in S}\psi_i[x]) \end{array}\right)\right) \\
\Leftrightarrow\ & \bigvee_{S\subseteq\{1,\ldots,m+1\}} \Big(\bigwedge_{i\in S}\phi_i \wedge \forall x.\Big(\psi_0[x] \vee \bigvee_{i\in S}\psi_i[x]\Big)\Big)
\end{aligned}$$

(*) holds because of:

$$\begin{aligned}
& \forall x.(a[x] \vee b[x]) \wedge \forall x.(a[x] \vee c) \\
\Leftrightarrow\ & \forall x.(a[x] \vee b[x]) \wedge (c \vee \forall x.a[x]) \\
\Leftrightarrow\ & (\forall x.(a[x] \vee b[x]) \wedge c) \vee (\forall x.(a[x] \vee b[x]) \wedge \forall x.a[x]) \\
\Leftrightarrow\ & (\forall x.(a[x] \vee b[x]) \wedge c) \vee \forall x.a[x]
\end{aligned}$$

Proof (Lem. 19). We show the conjecture by an induction over the subtrees of P. In the proof leaves, the hypothesis coincides with the assumption how CLOSE is applied in P_1, P_2. Otherwise, pick a subproof R (and the corresponding subproofs R_1, R_2 of P_1, P_2) and assume that the hypothesis holds for the direct subproofs of R. There are the following cases, depending on the constraint transformation that is performed by the rule applied in the root of R:

- The constraint is not changed (rules AND-LEFT, etc.): trivial
- Conjunction of constraints (rules AND-RIGHT, etc.): Let D^1, D^2 be the constraints of the direct subproofs of R_1. The constraints of the direct subproofs of R_2 are equivalent to $D^1 \vee \bigvee_{i=1}^{n_1} \phi_i^1$ and $D^2 \vee \bigvee_{i=1}^{n_2} \phi_i^2$. Then the constraint of R_1 is $D^1 \wedge D^2$ and the constraint of R_2 is equivalent to:

$$\begin{aligned} &\big(D^1 \vee \bigvee_{i=1}^{n_1} \phi_i^1\big) \wedge \big(D^2 \vee \bigvee_{i=1}^{n_2} \phi_i^2\big) \\ \Leftrightarrow\ &(D^1 \wedge D^2) \vee \bigvee_{i=1}^{n_1} (\phi_i^1 \wedge D^2) \vee \bigvee_{i=1}^{n_2} (\phi_i^2 \wedge D^1) \vee \bigvee_{i=1}^{n_1} \bigvee_{j=1}^{n_2} (\phi_i^1 \wedge \phi_j^2) \end{aligned}$$

- The rule COL-RED-SUBST applies a substitution to a constraint: let $[x/c']D_1$ be the constraint of the direct subproof of R_1 and $[x/c-u]D_1$ the constraint of R_1. The constraint $[x/c']D_2$ of the direct subproof of R_2 is then equivalent to $[x/c']D_1 \vee \bigvee_{i=1}^{n} \phi_i$, where each ϕ_i is shielded by Q, and the constraint of R_2 is equivalent to $[x/c-u]D_1 \vee \bigvee_{i=1}^{n} [c'/c-u]\phi_i$.
  - (i) $[x/c']D_1$ does not contain Q_c-constants. If $[x/c-u]D_1$ contains Q_c-constants, then also $c-u$ does and x occurs free in D_1. Because of the definition of free constant sets, then also $c' \in Q_c$ and then $[x/c']D_1$ contains Q_c-constants: contradiction.
  - (ii) We can assume that each ϕ_i has the form $\beta e + t \doteq 0 \wedge \psi$ with $e \in Q$, such that $d \prec_P e$ for all constants d in t. Due to the definition of $\prec_P$ we have $e \prec_P c'$ and thus $e \neq c'$ and c' does not occur in t. This implies that $[c'/c-u]\phi_i$ is shielded by Q:

$$[c'/c-u]\phi_i \quad \Leftrightarrow \quad \beta e + t \doteq 0 \wedge [c'/c-u]\psi$$

- The rule DIV-CLOSE' is applied: let $D_1 \Leftrightarrow [x/\alpha c']D_1'$ be the constraint of the direct subproof of R_1 and $[x/t]D_1' \vee \alpha \nmid t$ the constraint of R_1. Because D_1 does not contain any Q_c-constants, we can assume that D_1' does neither. The constraint of the direct subproof of R_2 is equivalent to $[x/\alpha c']D_1' \vee \bigvee_{i=1}^{n} \phi_i$, where each ϕ_i is shielded by Q.
  - (i) Because t does not contain Q_c-constants, neither does $[x/t]D_1' \vee \alpha \nmid t$.
  - (ii) We can assume that each ϕ_i has the form $\beta_i e_i + t_i \doteq 0 \wedge \psi_i$ with $e_i \in Q$. As for COL-RED-SUBST, it follows that c' does not occur in $\beta_i e_i + t_i$. Assume that $\psi_i \Leftrightarrow [x/\alpha c']\psi_i'$, then the formula $\beta_i e_i + t_i \doteq 0 \wedge [x/t]\psi_i'$ is shielded by Q. Altogether, the constraint of R_2 is equivalent to:

$$[x/t]D_1' \vee \alpha \nmid t \vee \bigvee_{i=1}^{n} (\beta_i e_i + t_i \doteq 0 \wedge [x/t]\psi_i')$$

- Existential quantification of constraints (rules EX-RIGHT, etc.): let $[x/c]D_1$ be the constraint of the direct subproof of R_1 and $\exists x.D_1$ the constraint of R_1. The constraint of the direct subproof of R_2 is equivalent to $[x/c]D_1 \vee \bigvee_{i=1}^{n} \phi_i$, where each ϕ_i is shielded by Q.

(i) Because $[x/c]D_1$ does not contain Q_c-constants, neither does $\exists x.D_1$.

(ii) The constraint of R_2 is equivalent to:

$$\exists x.\Big(D_1 \vee \bigvee_{i=1}^{n} [c/x]\phi_i\Big) \;\Leftrightarrow\; \exists x.D_1 \vee \bigvee_{i=1}^{n} \exists c.\phi_i$$

We can assume that each ϕ_i has the form $\beta e + t \doteq 0 \wedge \psi$ with $e \in Q$. As for COL-RED-SUBST, it follows that c does not occur in $\beta e + t$, and thus $\exists c.\phi_i \Leftrightarrow \beta e + t \doteq 0 \wedge \exists c.\psi$ is shielded by Q. By renaming it can be achieved that no illegal constants occur in $\exists c.\psi$.

– Universal quantification of constraints (rules ALL-RIGHT, etc.): let $[x/c]D_1$ be the constraint of the direct subproof of R_1 and $\forall x.D_1$ the constraint of R_1. The constraint of the direct subproof of R_2 is equivalent to $[x/c]D_1 \vee \bigvee_{i=1}^{n} \phi_i$, where each ϕ_i is shielded by Q.

(i) As for existential quantification.

(ii) We can assume that each ϕ_i has the form $t_i \doteq 0 \wedge \psi_i$, where $t_i \doteq 0$ is the shielding equation. Wlog., assume that $t_1, \ldots, t_k$ are the terms that contain c with a non-negative coefficient, while c does not occur in $t_{k+1}, \ldots, t_n$. This implies that c shields the formulae $\phi_1, \ldots, \phi_k$.

- If $c \notin Q$, it has to be the case that $k = 0$, as for COL-RED-SUBST. With the help of Lem. 27, we can rewrite the constraint of R_2 as follows:

$$\begin{aligned}
&\forall c.\Big([x/c]D_1 \vee \bigvee_{i=1}^{n}(t_i \doteq 0 \wedge \psi_i)\Big)\\
\Leftrightarrow\ &\bigvee_{S\subseteq\{1,\ldots,n\}} \Big(\bigwedge_{i\in S} t_i \doteq 0 \wedge \forall c.\Big([x/c]D_1 \vee \bigvee_{i\in S}\psi_u\Big)\Big)\\
\Leftrightarrow\ &\forall x.D_1 \vee \bigvee_{\substack{S\subseteq\{1,\ldots,n\}\\ S\neq\emptyset}} \Big(\bigwedge_{i\in S} t_i \doteq 0 \wedge \forall c.\Big([x/c]D_1 \vee \bigvee_{i\in S}\psi_u\Big)\Big)
\end{aligned}$$

- If $c \in Q$, then D_1 does not contain x by the induction hypothesis. We can again use Lem. 27 as follows:

$$\begin{aligned}
&\forall c.\Big(\underbrace{D_1 \vee \bigvee_{i=1}^{k}(t_i \doteq 0 \wedge \psi_i)}_{\psi_0[c]} \vee \bigvee_{i=k+1}^{n}(t_i \doteq 0 \wedge \psi_i)\Big)\\
\Leftrightarrow\ &\bigvee_{S\subseteq\{k+1,\ldots,n\}} \Big(\bigwedge_{i\in S} t_i \doteq 0 \wedge \forall c.\Big(\psi_0[c] \vee \bigvee_{i\in S}\psi_u\Big)\Big)\\
\Leftrightarrow\ &\forall c.\psi_0[c] \vee \bigvee_{\substack{S\subseteq\{k+1,\ldots,n\}\\ S\neq\emptyset}} \Big(\bigwedge_{i\in S} t_i \doteq 0 \wedge \forall c.\Big(\psi_0[c] \vee \bigvee_{i\in S}\psi_u\Big)\Big)
\end{aligned}$$

The formula $\forall c.\psi_0[c]$ can be simplified because all but the first disjunct are shielded by c:

$$\begin{aligned}\forall c.\Big(D_1 \vee \bigvee_{i=1}^{k}(t_i \doteq 0 \wedge \psi_i)\Big) &\Leftrightarrow D_1 \vee \forall c.\bigvee_{i=1}^{k}(t_i \doteq 0 \wedge \psi_i) \\ &\Leftrightarrow D_1 \Leftrightarrow \forall x.D_1\end{aligned}$$

In both cases, renaming can be used afterwards to eliminate c.