Verifying Mixed Real-Integer Quantifier Elimination

Amine Chaieb

Institut für Informatik
Technische Universität München

Abstract. We present a formally verified quantifier elimination procedure for the first order theory over linear mixed real-integer arithmetics in higher-order logic based on a work by Weispfenning. To this end we provide two verified quantifier elimination procedures: for Presburger arithmitics and for linear real arithmetics.

1 Introduction

The interest of theorem provers in decision procedures (dps.) for arithmetics is inveterate. Noteworthily, the apparently first theorem prover [14] implements a quantifier elimination procedure (qep.) for Presburger arithmetic ($\mathcal{Z}$). This paper presents a formally verified qep. for $\mathcal{R}_{\lfloor\cdot\rfloor} = \mathrm{Th}(\mathbb{R}, <, +, \lfloor.\rfloor, 0, 1)$ in higher-order logic based on [38]. For a real number x, $\lfloor x \rfloor$ is the greatest integer less than or equal to x. Our development environment is Isabelle/HOL [27]. Weispfenning presented in [38] a qep. for $\mathcal{R}_{\lfloor\cdot\rfloor}$, which reduces the qe. problem to qe. in $\mathcal{Z}$ and $\mathcal{R} = \mathrm{Th}(\mathbb{R}, <, +, 0, 1)$. In this paper, we formalize not only this reduction, but also a qep. for $\mathcal{Z}$ and a qep. for $\mathcal{R}$, which yields a complete qep. for $\mathcal{R}_{\lfloor\cdot\rfloor}$. In fact, our formalization is carried out in an executable fragment of HOL, for which code generation [7] is possible. The interest in $\mathcal{R}_{\lfloor\cdot\rfloor}$ is not only of theoretical nature (almost any non trivial extension of $\mathcal{R}_{\lfloor\cdot\rfloor}$ is undecidable, see [38] for several impossibility results), but also practically motivated, since mixed real-integer constraints naturally rise in verification.

Generated ML code from HOL functions [7] yields smoothly integratable oracles returning sound answers, provided the code generator is correct. Accepting these answers as equality proofs is often referred to by *reflection*. Many type theory based theorem provers accept such proofs as part of their underlying logic [21,8]. Reflection has been used and studied by many researchers and the opinions range from enthusiasm [2,3] to scepticism concerning its utility in LCF frameworks[19].

Regardless of reflection, implementations of dps. proved correct in the logic are worthy for several reasons: (a) while even new considerations using dependent types [1,23] fail to guarantee completeness of (complex enough) dps., the approach we adopt does; (b) sharing theorems between HOL theorem provers [29,26] provides a mechanism of sharing dps., which is an important issue to achieve faster progress in theorem proving; (c) an LCF-conservative integration is *still* possible, e.g. by specializing the simplifier to the involved defining equations (fast rewriting techniques [4] play an important role) or by instrumenting code generators to produce LCF-proofs (Isabelle provides a prototypical implementation [6]); (d) no intimate knowledge of the internals of the underlying theorem prover is needed. This makes the dps. easily portable and enables other developers (and even normal users) insight into dps.-implementations, which are the till now arcane.

The main contributions of our work are: (a) the first-time verified formalization of a qep. for $\mathcal{R}_{\lfloor\cdot\rfloor}$ and $\mathcal{R}$ (à la [15]) in a theorem prover; (b) a perspicuous and concise formalization (4000 lines including several optimizations) of a uniform treatment for linear arithmetics that is easily portable to other theorem provers; (c) the most substantial application of reflection in a theorem prover (as far as we are aware of); (d) a motivation to proof-producing code generators as an alternative of reflection in LCF-based theorem provers.

Related Work. In [38] $\mathcal{R}_{\lfloor\cdot\rfloor}$ has been proved to admit quantifier elimination. The overall procedure reduces the problem to qe. in $\mathcal{R}$ and $\mathcal{Z}$. The decidability of $\mathcal{R}$ is arguably due to Fourier [18]. The decidability of $\mathcal{Z}$ has been shown independently by Presburger [31] and Skolem [34]. Several other qep. have been proposed for $\mathcal{R}$ [35,15,24] and for $\mathcal{Z}$ [12,33,32] and incited excellent complexity studies [30,17,16,36,37]. Alternative dps. use automata [39,22]. Many theorem provers include implementations of qep. for $\mathcal{Z}$ and $\mathcal{R}$ [28,10,25] or for some subsets [13]. A formalization of Cooper's qep. for $\mathcal{Z}$ has been presented in [11]. An automata based dp. for *closed* $\mathcal{R}_{\lfloor\cdot\rfloor}$-formulae has been recently proposed [9]. Since this procedure is based on sorts distinction for variables, it does *not* provide a qep., see [38] for an excellent proof. An extension of [32] to deal with real and integer variables is presented in [5]. The said extension is noteworthily online and proof producing and hence useful for combination frameworks based on the Nelson and Oppen method. The considered formulae are *quantifier free*.

Notation. Datatypes are declared using datatype. Lists are built up from the empty list [] and consing $\cdot$; the infix @ appends two lists. For a list l, $\{\!\{l\}\!\}$ denotes the set of elements of l, and $l!n$ denotes its n^{th} element. Functions are defined by pattern matching. We use the letters u, x, y, z for reals and c, d, i, j for integers and denote by $\underline{i}$ the injection of i into the reals. We call $\underline{i}$ a real integer. For i and j we write $i \mid j$ if i divides j. For x and y we define $x \mid y \leftrightarrow \exists i. y = x{\cdot}\underline{i}$. We use $i \nmid j$ (resp. $x \nmid y$) as a shorthand for $\neg i \mid j$ (resp. $\neg x \mid y$). For x we denote by $\lfloor x\rfloor$ the greatest integer i such that $\underline{i} \leq x$. Note that $\lfloor x\rfloor$ is an integer, not a real integer. We use $\lceil x\rceil$ as a shorthand for $-\lfloor -x\rfloor$.

The rest of this paper is structured as follows. In § 2 we set up the basis for our formalization and then present the qep. for $\mathcal{R}_{\lfloor\cdot\rfloor}$ in a top-down fashion. In § 3 we formalize the overall procedure in terms of two qe. procedures: for $\mathcal{R}$, subject of § 4, and for $\mathcal{Z}$, subject of § 5. In § 6 we describe some formalization and integration issues.

2 Preliminaries

2.1 Syntax and Semantics

We define the syntax of terms and formulae as follows:

$$\begin{array}{l} \mathsf{datatype}\ \rho = \widehat{int} \mid \boldsymbol{v}_{nat} \mid \boldsymbol{-}\ \rho \mid \rho \boldsymbol{+} \rho \mid \rho \boldsymbol{-} \rho \mid int \boldsymbol{*} \rho \mid \lfloor \rho \rfloor \\ \mathsf{datatype}\ \phi = \rho \boldsymbol{<0} \mid \rho \boldsymbol{>0} \mid \rho \boldsymbol{\leq 0} \mid \rho \boldsymbol{\geq 0} \mid \rho \boldsymbol{=0} \mid \rho \boldsymbol{\neq 0} \mid int \mid \rho \mid int \nmid \rho \\ \qquad\qquad \boldsymbol{T} \mid \boldsymbol{F} \mid \neg\ \phi \mid \phi \wedge \phi \mid \phi \vee \phi \mid \phi \rightarrow \phi \mid \phi \leftrightarrow \phi \mid \exists\, \phi \mid \forall\, \phi \end{array}$$

The real integer constant $\underline{i}$ in the logic is represented by the term $\widehat{i}$. Bound variables are represented by de Bruijn indices: $\boldsymbol{v}_n$ represents the bound variable with index n (a natural number). Hence quantifiers need not carry variable names. The bold symbols $\boldsymbol{+}$, $\boldsymbol{\leq 0}$, $\wedge$ etc are constructors and reflect their counterparts $+$, $\lambda x.x \leq 0$, $\wedge$ etc in the logic. We use $\bowtie \boldsymbol{0}$ as a place-holder for $\boldsymbol{=0}, \boldsymbol{\neq 0}, \boldsymbol{<0}, \boldsymbol{\leq 0}, \boldsymbol{>0}$ or $\boldsymbol{\geq 0}$, all of them notably constructors with *only one* argument. We use $\lceil t \rceil$ to denote $\boldsymbol{-}\ \lfloor \boldsymbol{-}\ t \rfloor$.

Throughout the paper p and q (resp. s and t) are of type ϕ (resp. ρ).

$$\begin{array}{lllllllll}
\llbracket \widehat{i} \rrbracket_\rho^{vs} & = \underline{i} & & \llbracket \boldsymbol{T} \rrbracket^{vs} & = True & & \llbracket i \mid t \rrbracket^{vs} & = (\underline{i} \mid \llbracket t \rrbracket_\rho^{vs}) \\
\llbracket \boldsymbol{v}_n \rrbracket_\rho^{vs} & = vs!n & & \llbracket \boldsymbol{F} \rrbracket^{vs} & = False & & \llbracket i \nmid t \rrbracket^{vs} & = (\underline{i} \nmid \llbracket t \rrbracket_\rho^{vs}) \\
\llbracket \boldsymbol{-}\ t \rrbracket_\rho^{vs} & = -\llbracket t \rrbracket_\rho^{vs} & & \llbracket t \boldsymbol{<0} \rrbracket^{vs} & = (\llbracket t \rrbracket_\rho^{vs} < \underline{0}) & & \llbracket \neg p \rrbracket^{vs} & = (\neg \llbracket p \rrbracket^{vs}) \\
\llbracket t \boldsymbol{+} s \rrbracket_\rho^{vs} & = \llbracket t \rrbracket_\rho^{vs} + \llbracket s \rrbracket_\rho^{vs} & & \llbracket t \boldsymbol{>0} \rrbracket^{vs} & = (\llbracket t \rrbracket_\rho^{vs} > \underline{0}) & & \llbracket p \wedge q \rrbracket^{vs} & = (\llbracket p \rrbracket^{vs} \wedge \llbracket q \rrbracket^{vs}) \\
\llbracket t \boldsymbol{-} s \rrbracket_\rho^{vs} & = \llbracket t \rrbracket_\rho^{vs} - \llbracket s \rrbracket_\rho^{vs} & & \llbracket t \boldsymbol{\leq 0} \rrbracket^{vs} & = (\llbracket t \rrbracket_\rho^{vs} \leq \underline{0}) & & \llbracket p \vee q \rrbracket^{vs} & = (\llbracket p \rrbracket^{vs} \vee \llbracket q \rrbracket^{vs}) \\
\llbracket i \boldsymbol{*} t \rrbracket_\rho^{vs} & = \underline{i} \cdot \llbracket t \rrbracket_\rho^{vs} & & \llbracket t \boldsymbol{\geq 0} \rrbracket^{vs} & = (\llbracket t \rrbracket_\rho^{vs} \geq \underline{0}) & & \llbracket p \rightarrow q \rrbracket^{vs} & = (\llbracket p \rrbracket^{vs} \rightarrow \llbracket q \rrbracket^{vs}) \\
\llbracket \lfloor t \rfloor \rrbracket_\rho^{vs} & = \underline{\lfloor \llbracket t \rrbracket_\rho^{vs} \rfloor} & & \llbracket t \boldsymbol{=0} \rrbracket^{vs} & = (\llbracket t \rrbracket_\rho^{vs} = \underline{0}) & & \llbracket p \leftrightarrow q \rrbracket^{vs} & = (\llbracket p \rrbracket^{vs} \leftrightarrow \llbracket q \rrbracket^{vs}) \\
 & & & \llbracket t \boldsymbol{\neq 0} \rrbracket^{vs} & = (\llbracket t \rrbracket_\rho^{vs} \neq \underline{0}) & & \llbracket \exists\, p \rrbracket^{vs} & = (\exists x. \llbracket p \rrbracket^{x \cdot vs}) \\
 & & & & & & \llbracket \forall\, p \rrbracket^{vs} & = (\forall x. \llbracket p \rrbracket^{x \cdot vs})
\end{array}$$

Fig. 1. Semantics of the shadow syntax

The interpretation functions ($\llbracket . \rrbracket_\rho^{\cdot}$ and $\llbracket . \rrbracket^{\cdot}$) in Fig. 1 map the representations back into logic. They are parameterized by an environment vs which is a list of real expressions. The de Bruijn index $\boldsymbol{v}_n$ picks out the n^{th} element from that list. We say that x is a witness for p if $\llbracket p \rrbracket^{x \cdot vs}$ holds. It will alway be clear from the context which vs is meant.

2.2 Generic Quantifier Elimination

Assume we have a function qe, that eliminates one $\exists$ in front of quantifier-free formulae. The function qelim_ϕ applies qe to all quantified subformulae in a bottom-up fashion. Let $\mathsf{qfree}\ p$ formalize that p is quantifier-free (qf.). We prove by structural induction that if qe takes a qf. formula q and returns a qf. formula q' equivalent to $\exists\, q$, then $\mathsf{qelim}_\phi\ qe$ is a qep.:

$$\begin{array}{c} (\forall vs, q.\ \mathsf{qfree}\ q \rightarrow \mathsf{qfree}\ (qe\ q) \wedge (\llbracket qe\ q \rrbracket^{vs} \leftrightarrow \llbracket \exists\, q \rrbracket^{vs})) \\ \rightarrow \mathsf{qfree}\ (\mathsf{qelim}_\phi\ qe\ p) \wedge (\llbracket \mathsf{qelim}_\phi\ qe\ p \rrbracket^{vs} \leftrightarrow \llbracket p \rrbracket^{vs}). \end{array} \tag{1}$$

In § 3 we present mir, an instance of qe satisfying the premise of (1).

2.3 Linearity

When defining a function (over ρ or ϕ) we assume the input to have a precise syntactical shape. This not only simplifies the function definition but is also crucial for its correctness proof. The fact that $\boldsymbol{v}_0$ does not occur in a ρ-term t (resp. in a ϕ-formula p) is formalized by $\mathsf{unbound}_\rho\ t$ (resp. $\mathsf{unbound}_\phi\ p$). Substituting t for $\boldsymbol{v}_0$ in p is defined by $p[t]$. Decreasing all variable indexes in p is defined by $\mathsf{decr}\ p$. These functions have such simple recursive definitions that the properties (2) are proved automatically.

$$\begin{gathered}\mathsf{unbound}_\phi\ p \rightarrow \forall x, y. [\![p]\!]^{x \cdot vs} \leftrightarrow [\![p]\!]^{y \cdot vs} \\ \mathsf{qfree}\ p \rightarrow ([\![p[t]]\!]^{x \cdot vs} \leftrightarrow [\![p]\!]^{([\![t]\!]_\rho^{x \cdot vs}) \cdot vs}) \\ \mathsf{unbound}_\phi\ p \rightarrow \forall x. [\![\mathsf{decr}\ p]\!]^{vs} \leftrightarrow [\![p]\!]^{x \cdot vs}\end{gathered} \tag{2}$$

We define p to be $\mathcal{R}$-linear ($\mathsf{islin}_\mathcal{R}\ p$) if it is built up from $\wedge, \vee$ and atoms θ, either of the form $c * \boldsymbol{v}_0 \boldsymbol{+} t \bowtie \boldsymbol{0}$, such that $\mathsf{unbound}_\rho\ t \wedge c > 0$, or satisfying $\mathsf{unbound}_\phi\ \theta$. We define p to be $\mathcal{Z}$-linear in a context vs ($\mathsf{islin}_\mathcal{Z}\ p\ vs$) if in addition to the previous requirements every t represents an integer, i.e. $[\![\lfloor t \rfloor]\!]_\rho^{vs} = [\![t]\!]_\rho^{vs}$. Moreover $i \mid c * \boldsymbol{v}_0 \boldsymbol{+} t$ and $i \nmid c * \boldsymbol{v}_0 \boldsymbol{+} t$ such that $i > 0 \wedge c > 0 \wedge \mathsf{unbound}_\rho\ t \wedge [\![t]\!]_\rho^{vs} = [\![\lfloor t \rfloor]\!]_\rho^{vs}$, are $\mathcal{Z}$-linear atoms. A $\mathcal{R}$- (resp. $\mathcal{Z}$-) linear formula can be regarded as a formula in $\mathcal{R}$ (resp. $\mathcal{Z}$), assuming $\boldsymbol{v}_0$ will be interpreted by some $x \in \mathbb{R}$ (resp. by some $\underline{i}, i \in \mathbb{Z}$).

3 Quantifier Elimination for $\mathcal{R}_{\lfloor \cdot \rfloor}$

The main idea is: "$\lfloor \cdot \rfloor$ *is burdensome, get rid of it*". Notice that $\forall x. \underline{0} \leq x - \lfloor x \rfloor < \underline{1}$ and hence $\exists x. [\![p]\!]^{x \cdot vs} \leftrightarrow \exists i, u. \underline{0} \leq u < \underline{1} \wedge [\![p]\!]^{(\underline{i}+u) \cdot vs}$. Let $\widehat{0} \mathrel{\underline{\boldsymbol{\leq}}} \boldsymbol{v}_0 \boldsymbol{<} \widehat{1}$ be a shorthand for $1 * \boldsymbol{v}_0 \boldsymbol{+} \widehat{0} \mathrel{\underline{\boldsymbol{\geq}}} \boldsymbol{0} \wedge 1 * \boldsymbol{v}_0 \boldsymbol{+} \widehat{-1} \boldsymbol{<} \boldsymbol{0}$. Let $\mathsf{split}_0\ p = \widehat{0} \mathrel{\underline{\boldsymbol{\leq}}} \boldsymbol{v}_0 \boldsymbol{<} \widehat{1} \wedge p'$, where p' results from p by replacing every occurrence of $\boldsymbol{v}_0$ by $\boldsymbol{v}_0 \boldsymbol{+} \boldsymbol{v}_1$ and $\boldsymbol{v}_i$ by $\boldsymbol{v}_{i+1}$ for $i > 0$. We easily prove

$$\mathsf{qfree}\ p \rightarrow ([\![\exists\, p]\!]^{vs} \leftrightarrow \exists i, u. [\![\mathsf{split}_0\ p]\!]^{\underline{i} \cdot u \cdot vs}) \tag{3}$$

One main contribution of [38] is to supply two functions $\mathsf{lin}_\mathcal{R}$ and $\mathsf{lin}_\mathcal{Z}$, which, assuming that $\boldsymbol{v}_0$ is interpreted by $u \in [\underline{0}, \underline{1})$ (resp. by $\underline{i}$), transform any qf. p into a $\mathcal{R}$- (resp. $\mathcal{Z}$-) linear formula, cf. (5) and (4).

$$\mathsf{qfree}\ p \rightarrow ([\![\mathsf{lin}_\mathcal{Z}\ p]\!]^{\underline{i} \cdot vs} \leftrightarrow [\![p]\!]^{\underline{i} \cdot vs}) \wedge \mathsf{islin}_\mathcal{Z}\ (\mathsf{lin}_\mathcal{Z}\ p)\ (\underline{i} \cdot vs) \tag{4}$$

$$\mathsf{qfree}\ p \wedge \underline{0} \leq x < \underline{1} \rightarrow ([\![\mathsf{lin}_\mathcal{R}\ p]\!]^{x \cdot vs} \leftrightarrow [\![p]\!]^{x \cdot vs}) \wedge \mathsf{islin}_\mathcal{R}\ (\mathsf{lin}_\mathcal{R}\ p) \tag{5}$$

The next subsections exhibit $\mathsf{lin}_\mathcal{Z}$ and $\mathsf{lin}_\mathcal{R}$, which mainly *"get rid of* $\lfloor \cdot \rfloor$ *"*. Now given two qe. procedures $qe_{\mathcal{R}_l}$ for $\mathcal{R}$ and $qe_{\mathcal{Z}_l}$ for $\mathcal{Z}$ satisfying:

$$\begin{aligned}\mathsf{islin}_\mathcal{R}\ p \rightarrow ([\![qe_{\mathcal{R}_l}\ \widehat{0} \mathrel{\underline{\boldsymbol{\leq}}} \boldsymbol{v}_0 \boldsymbol{<} \widehat{1} \wedge p]\!]^{vs} \leftrightarrow [\![\exists\, \widehat{0} \mathrel{\underline{\boldsymbol{\leq}}} \boldsymbol{v}_0 \boldsymbol{<} \widehat{1} \wedge p]\!]^{vs}) \wedge \mathsf{qfree}(qe_{\mathcal{R}_l}\ p) \\ \mathsf{islin}_\mathcal{Z}\ p \rightarrow ([\![qe_{\mathcal{Z}_l}\ p]\!]^{vs} \leftrightarrow \exists i. [\![p]\!]^{\underline{i} \cdot vs}) \wedge \mathsf{qfree}(qe_{\mathcal{Z}_l}\ p)\end{aligned}$$

then it is simple to prove that $\mathsf{mir} = qe_{\mathcal{Z}_l} \circ \mathsf{lin}_\mathcal{Z} \circ qe_{\mathcal{R}_l} \circ \mathsf{lin}_\mathcal{R} \circ \mathsf{split}_0$ satisfies the premise of (1) and hence $\mathsf{qelim}_\phi\ \mathsf{mir}$ is a qep. for ϕ-formulae. In § 4 and § 5 we present instances of $qe_\mathcal{R}$ and $qe_\mathcal{Z}$.

3.1 $\mathsf{lin}_{\mathcal{Z}}$

In order to define $\mathsf{lin}_{\mathcal{Z}}$ and prove (4), we first introduce a function $\mathsf{split}_{\mathcal{Z}}$ that, given a ρ-term t, returns an integer c and a ρ-term s (not involving $\boldsymbol{v}_0$), such that (6) (Lemma 3.2 in [38]) holds. Note that $\boldsymbol{v}_0$ is interpreted by a real integer $\underline{i}$.

$$(\mathsf{split}_{\mathcal{Z}}\ t = (c,s)) \rightarrow (\llbracket c * \boldsymbol{v}_0 + s\rrbracket_\rho^{\underline{i}\cdot vs} = \llbracket t\rrbracket_\rho^{\underline{i}\cdot vs}) \wedge \mathsf{unbound}_\rho s \tag{6}$$

The definition of $\mathsf{split}_{\mathcal{Z}}$ and the proof of (6) proceed by induction on t. If $t = \lfloor t'\rfloor$ then return $(c, \lfloor s\rfloor)$, where $\mathsf{split}_{\mathcal{Z}}\ t' = (c,s)$. Remember that $\lfloor x + \underline{j}\rfloor = \lfloor x\rfloor + j$ holds for any $j \in \mathbb{Z}$. The other cases are trivial.

Now $\mathsf{lin}_{\mathcal{Z}}$ is simple: push negations inwards and transform atoms according to the result of $\mathsf{split}_{\mathcal{Z}}$ and the properties (7), cf. example 1 for the $\boldsymbol{=0}$ case, where the first property in (7) is used . By induction on p, we easily prove (4) using the properties (6) and (7).

$$\begin{aligned}(\underline{c{\cdot}i} = y) &\leftrightarrow (c{\cdot}i = \lfloor y\rfloor \wedge y = \underline{\lfloor y\rfloor})\\ (\underline{c{\cdot}i} < y) &\leftrightarrow (c{\cdot}i < \lfloor y\rfloor \vee (c{\cdot}i = \lfloor y\rfloor \wedge \underline{\lfloor y\rfloor} < y))\\ (\underline{d} \mid \underline{c{\cdot}i} + y) &\leftrightarrow (\underline{\lfloor y\rfloor} = y \wedge d \mid c{\cdot}i + \lfloor y\rfloor)\\ \underline{0} \mid x &\leftrightarrow (x = \underline{0})\end{aligned} \tag{7}$$

Example 1

```
$\mathit{lin}_{\mathcal{Z}}\ (t \boldsymbol{=0}) = \boldsymbol{let}\ (c,s) = \mathit{split}_{\mathcal{Z}}\ t\ \boldsymbol{in}$
                 $\boldsymbol{if}\ c = 0\ \boldsymbol{then}\ s \boldsymbol{=0}$
                 $\boldsymbol{else\ if}\ c > 0\ \boldsymbol{then}\ c * \boldsymbol{v}_0 + \lceil s\rceil \boldsymbol{=0} \wedge \lfloor s\rfloor - s \boldsymbol{=0}$
                 $\boldsymbol{else}\ -c * \boldsymbol{v}_0 + \lfloor -s\rfloor \boldsymbol{=0} \wedge \lfloor s\rfloor - s \boldsymbol{=0}$
```

3.2 $\mathsf{lin}_{\mathcal{R}}$

In order to define $\mathsf{lin}_{\mathcal{R}}$ and prove (5), we first introduce a function $\mathsf{split}_{\mathcal{R}} : \rho \rightarrow (\phi \times \mathit{int} \times \rho)\mathit{list}$ that, given a ρ-term t, yields a *complete* finite case distinction given by $\mathcal{R}$-linear formulae ϕ_i and corresponding ρ-terms s_i (not involving $\boldsymbol{v}_0$) and integers c_i such that $\llbracket t\rrbracket_\rho^{u\cdot vs} = \llbracket c_i * \boldsymbol{v}_0 + s_i\rrbracket_\rho^{u\cdot vs}$ whenever $\llbracket\phi_i\rrbracket^{u\cdot vs}$ holds (Lemma 3.3 in [38]), i.e.

$$\begin{aligned}\forall(\phi_i, c_i, s_i) \in \{\!\{\mathsf{split}_{\mathcal{R}}\ t\}\!\}.(\llbracket\phi_i\rrbracket^{u\cdot vs} \rightarrow (\llbracket t\rrbracket_\rho^{u\cdot vs} = \llbracket c_i * \boldsymbol{v}_0 + s_i\rrbracket_\rho^{u\cdot vs}))&\\ \wedge \mathsf{unbound}_\rho\ s_i \wedge \mathsf{islin}_{\mathcal{R}}\ \phi_i& \quad (8)\\ \underline{0} \le u < \underline{1} \rightarrow \exists(\phi_i, c_i, s_i) \in \{\!\{\mathsf{split}_{\mathcal{R}}\ t\}\!\}.\llbracket\phi_i\rrbracket^{u\cdot vs}& \quad (9)\end{aligned}$$

The definition of $\mathsf{split}_{\mathcal{R}}$ and the proof of (8) and (9) proceed by induction on t. Assume $t = \lfloor t'\rfloor$, let $(\phi_i', c_i', s_i') \in \{\!\{\mathsf{split}_{\mathcal{R}}\ t'\}\!\}$ and assume $\llbracket\phi_i'\rrbracket^{u\cdot vs}$ and $c_i' > 0$. From the induction hypothesis we have $\llbracket t'\rrbracket_\rho^{u\cdot vs} = \llbracket c_i' * \boldsymbol{v}_0 + s_i'\rrbracket_\rho^{u\cdot vs}$ and since $\underline{0} \le u < \underline{1}$ and $c_i' > 0$ we have $j \le c_i'{\cdot}u < j + 1$ for some $j \in \{0 \dots c_i'\}$, i.e.

$$\underline{j + \lfloor\llbracket s_i'\rrbracket_\rho^{u\cdot vs}\rfloor} \le \llbracket c_i' * \boldsymbol{v}_0 + s_i'\rrbracket_\rho^{u\cdot vs} < \underline{j + 1 + \lfloor\llbracket s_i'\rrbracket_\rho^{u\cdot vs}\rfloor}$$

and hence $\lfloor [\![c'_i * \boldsymbol{v}_0 + s'_i]\!]^{u \cdot vs}_{\rho} \rfloor = j$. For $(\phi'_i, c'_i, s'_i) \in \{\!\{\mathsf{split}_{\mathcal{R}}\ t'\}\!\}$ $\mathsf{split}_{\mathcal{R}}$ returns the list of $(\phi'_i \wedge A_j, 0, \lfloor s \rfloor + \widehat{j})$, where $j \in \{0 \ldots c'_i\}$, where $A_j = r - \widehat{j} \mathbin{\underline{\geq}} \mathbf{0} \wedge r - \widehat{j+1} \mathbin{<} \mathbf{0}$ and $r = c'_i * \boldsymbol{v}_0 + s'_i - \lfloor s'_i \rfloor$. The cases $c'_i < 0$ and $c'_i = 0$, ignored in [38], are analogous. The other cases for t are simple.

The definition of $\mathsf{lin}_{\mathcal{R}}$ for atoms is involved, but very simple for the rest: it just pushes negations inwards. Due to the result of $\mathsf{split}_{\mathcal{R}}$, assume that atoms have the form $f(c * \boldsymbol{v}_0 + s)$, where s does not involve $\boldsymbol{v}_0$ and $f \in \{\bowtie \mathbf{0}, \lambda t.i \mid t, \lambda t.i \nmid t$ for some $i\}$. For every f, we define its corresponding $\mathcal{R}$-*linear* version $f_l : int \to \rho \to \phi$, and prove (10). Example 2 shows the case for $= \mathbf{0}$ and the corresponding definition of $\mathsf{lin}_{\mathcal{R}}$.

$$\begin{aligned} \underline{0} \leq u < \underline{1} \wedge \mathsf{unbound}_{\rho}\ s \wedge ([\![t]\!]^{u \cdot vs}_{\rho} = [\![c * \boldsymbol{v}_0 + s]\!]^{u \cdot vs}_{\rho}) \\ \to ([\![f_l\ c\ s]\!]^{u \cdot vs} \leftrightarrow [\![f\ t]\!]^{u \cdot vs}) \wedge \mathsf{islin}_{\mathcal{R}}\ (f_l\ c\ s) \end{aligned} \tag{10}$$

Example 2

$$\begin{aligned} c * \boldsymbol{v}_0 + s =_l \mathbf{0} = \textit{\textbf{if}}\ c = 0\ \textit{\textbf{then}}\ s = \mathbf{0}\ \textit{\textbf{else}} \\ \textit{\textbf{if}}\ c > 0\ \textit{\textbf{then}}\ c * \boldsymbol{v}_0 + s = \mathbf{0}\ \textit{\textbf{else}}\ -c * \boldsymbol{v}_0 +- s = \mathbf{0} \end{aligned}$$

$$\begin{aligned} \mathit{lin}_{\mathcal{R}}(t = \mathbf{0}) = \textit{\textbf{let}}\ [(p_0, c_0, s_0), ..., (p_n, c_n, s_n)] = \mathit{split}_{\mathcal{R}}\ t \\ \textit{\textbf{in}}\ (p_0 \wedge (c_0 * \boldsymbol{v}_0 + s_0 =_l \mathbf{0})) \vee ... \vee (p_n \wedge (c_n * \boldsymbol{v}_0 + s_n =_l \mathbf{0})) \end{aligned}$$

Since $\cdot \mid \cdot$ and $\cdot \nmid \cdot$ are not $\mathcal{R}$-linear, their corresponding linear versions eliminate them at the cost of a case distinction according to (11).

$$\begin{aligned} \underline{0} \leq u < \underline{1} \wedge c > 0 \to \\ (\underline{d} \mid \underline{c} \cdot u + s \leftrightarrow \exists j \in \{0..c-1\}.(\underline{c} \cdot u = \underline{j} + \underline{\lceil s \rceil} - s) \wedge d \mid j + \lceil s \rceil) \end{aligned} \tag{11}$$

We implement this case distinction by dvd and $\cdot \mid_l \cdot$ follows naturally, ie.

$$\begin{aligned} d\ \mathsf{dvd}\ c * \boldsymbol{v}_0 + s &= (c * \boldsymbol{v}_0 + s - \lceil s \rceil - \widehat{0} = \mathbf{0} \wedge d \mid \lceil s \rceil + \widehat{c-1}) \vee ... \\ &\quad \vee (c * \boldsymbol{v}_0 + s - \lceil s \rceil - \widehat{c-1} = \mathbf{0} \wedge d \mid \lceil s \rceil + \widehat{c-1}) \\ d \mid_l c * \boldsymbol{v}_0 + s &= \ \ \mathbf{if}\ d = 0\ \mathbf{then}\ c * \boldsymbol{v}_0 + s =_l \mathbf{0}\ \mathbf{else} \\ &\quad\ \mathbf{if}\ c = 0\ \mathbf{then}\ d \mid s\ \mathbf{else} \\ &\quad\ \mathbf{if}\ c > 0\ \mathbf{then}\ |d|\ \mathsf{dvd}\ c * \boldsymbol{v}_0 + s \\ &\quad\ \mathbf{else\ then}\ |d|\ \mathsf{dvd}\ -c * \boldsymbol{v}_0 +- s \end{aligned}$$

Now we define $\mathsf{lin}_{\mathcal{R}}(d \mid t)$ analogously to the $= \mathbf{0}$-case.

$$\begin{aligned} \mathsf{lin}_{\mathcal{R}}(d \mid t) = \mathbf{let}\ [(p_0, c_0, s_0), ..., (p_n, c_n, s_n)] = \mathsf{split}_{\mathcal{R}}\ t \\ g = \lambda c, s.d \mid_l c * \boldsymbol{v}_0 + s \\ \mathbf{in}\ (p_0 \wedge (g\ c_0\ s_0)) \vee \ldots \vee (p_n \wedge (g\ c_n\ s_n)) \end{aligned}$$

Note that $\mathsf{lin}_{\mathcal{R}}$ has akin definitions for the atoms. In fact for an atom $f(t)$, the real definition is $\mathsf{lin}_{\mathcal{R}}(f(t)) = \mathsf{split}_l\ f_l\ t$, where

$$\begin{aligned} \mathsf{split}_l\ f_l\ t \equiv \mathbf{let}\ [(p_0, c_0, s_0), \ldots, (p_n, c_n, s_n)] = \mathsf{split}_{\mathcal{R}}\ t \\ \mathbf{in}\ (p_0 \wedge (f_l\ c_0\ s_0)) \vee \ldots \vee (p_n \wedge (f_l\ c_n\ s_n)) \end{aligned} \tag{12}$$

We prove the following simple, yet generic property for split_l

$$\begin{aligned}\underline{0} \leq u < \underline{1} \wedge (\forall t, c, s.\mathsf{unbound}_\rho\ s \wedge (\llbracket t \rrbracket_\rho^{u\cdot vs} = \llbracket c * \boldsymbol{v}_0 + s \rrbracket_\rho^{u\cdot vs})\\ \rightarrow (\llbracket f_l\ c\ s \rrbracket^{u\cdot vs} \leftrightarrow \llbracket f\ t \rrbracket^{u\cdot vs} \wedge \mathsf{islin}_{\mathcal{R}}\ (f_l\ c\ s)))\\ \rightarrow \mathsf{islin}_{\mathcal{R}}\ (\mathsf{split}_l\ f_l\ t) \wedge (\llbracket \mathsf{split}_l\ f_l\ t \rrbracket^{u\cdot vs} \leftrightarrow \llbracket f\ t \rrbracket^{u\cdot vs})\end{aligned} \tag{13}$$

Note that the premise of (13), which expresses that f_l is a $\mathcal{R}$-linear version of f, will be discharged by the instances of (10) for the different f's. After all these preparations, it is not surprising that (5) is proved automatically.

4 Quantifier Elimination for $\mathcal{R}$

We present $\mathsf{ferrack}$, a verified qep. for $\mathcal{R}$ based on [15], and prove (14). To our knowledge, this is the first-time verified formalization of this qep.

$$\mathsf{islin}_{\mathcal{R}}\ p \rightarrow \llbracket \mathsf{ferrack}(\widehat{0} \leq \boldsymbol{v}_0 < \widehat{1} \wedge p) \rrbracket^{vs} \leftrightarrow \llbracket \exists\, \widehat{0} \leq \boldsymbol{v}_0 < \widehat{1} \wedge p \rrbracket^{vs} \tag{14}$$

The implementation of $\mathsf{ferrack}$ is based on (15) (Lemma 1.1 in [15]), a consequence of the nature of $\mathcal{R}$-expressible sets: for a $\mathcal{R}$-linear formula p, the set $\{x | \llbracket p \rrbracket^{x\cdot vs}\}$ is a finite union of *intervals*, whose endpoints are either $\frac{\llbracket t \rrbracket_\rho^{x\cdot vs}}{\underline{c}}$ for some $(t, c) \in \{\!\{\mathcal{U}\ p\}\!\}$ (cf. Fig. 2), $-\infty$ or $+\infty$. In Fig. 2, p_- and p_+ are defined as

p	$\mathcal{U}\ p$	$\mathcal{B}\ p$	p_-	p_+
$p \wedge q$	$(\mathcal{U}\ p)@(\mathcal{U}\ q)$	$(\mathcal{B}\ p)@(\mathcal{B}\ q)$	$p_- \wedge q_-$	$p_+ \wedge q_+$
$p \vee q$	$(\mathcal{U}\ p)@(\mathcal{U}\ q)$	$(\mathcal{B}\ p)@(\mathcal{B}\ q)$	$p_- \vee q_-$	$p_+ \vee q_+$
$c * \boldsymbol{v}_0 + t = \boldsymbol{0}$	$[(-\ t, c)]$	$[\widehat{-1} -\ t]$	$\boldsymbol{F}$	$\boldsymbol{F}$
$c * \boldsymbol{v}_0 + t \neq \boldsymbol{0}$	$[(-\ t, c)]$	$[-\ t]$	$\boldsymbol{T}$	$\boldsymbol{T}$
$c * \boldsymbol{v}_0 + t < \boldsymbol{0}$	$[(-\ t, c)]$	$[]$	$\boldsymbol{T}$	$\boldsymbol{F}$
$c * \boldsymbol{v}_0 + t \leq \boldsymbol{0}$	$[(-\ t, c)]$	$[]$	$\boldsymbol{T}$	$\boldsymbol{F}$
$c * \boldsymbol{v}_0 + t > \boldsymbol{0}$	$[(-\ t, c)]$	$[-\ t]$	$\boldsymbol{F}$	$\boldsymbol{T}$
$c * \boldsymbol{v}_0 + t \geq \boldsymbol{0}$	$[(-\ t, c)]$	$[\widehat{-1} -\ t]$	$\boldsymbol{F}$	$\boldsymbol{T}$
_	$[]$	$[]$	p	p

Fig. 2. $\mathcal{U}\ p, \mathcal{B}\ p, p_-, p_+$

to simulate the behavior of p, where $\boldsymbol{v}_0$ is interpreted by arbitrarily small (resp. big) real numbers.

$$\begin{aligned}\mathsf{islin}_{\mathcal{R}}\ p \rightarrow (\exists x.\llbracket p \rrbracket^{x\cdot vs} \leftrightarrow \llbracket p_- \rrbracket^{x\cdot vs} \vee \llbracket p_+ \rrbracket^{x\cdot vs}\\ \vee \exists((t, i), (s, j)) \in \{\!\{\mathcal{U}\ p\}\!\}^2.\llbracket p \rrbracket^{((\llbracket t \rrbracket_\rho^{x\cdot vs}/\underline{i} + \llbracket s \rrbracket_\rho^{x\cdot vs}/\underline{j})/\underline{2})\cdot vs})\end{aligned} \tag{15}$$

For the proof of (15), assume $\mathsf{islin}_{\mathcal{R}}\ p$. The conclusion of (15) has the form $A \leftrightarrow B \vee C \vee D$. Obviously $D \rightarrow A$ holds. We first prove $B \rightarrow A$ and $C \rightarrow A$. For this we prove the following properties for p_- and p_+ by induction on p. The proof is simple: we provide y.

$$\mathsf{islin}_{\mathcal{R}}\ p \rightarrow \mathsf{unbound}_\phi\ (p_-) \wedge \exists y.\forall x < y.\llbracket p_- \rrbracket^{x\cdot vs} \leftrightarrow \llbracket p \rrbracket^{x\cdot vs} \tag{16}$$

$$\mathsf{islin}_{\mathcal{R}}\ p \rightarrow \mathsf{unbound}_\phi\ (p_+) \wedge \exists y.\forall x > y.\llbracket p_+ \rrbracket^{x\cdot vs} \leftrightarrow \llbracket p \rrbracket^{x\cdot vs} \tag{17}$$

Now assume that $[\![p_-]\!]^{x \cdot vs}$ holds for some x. Since $\mathsf{unbound}_\phi\ p_-$ holds, we have by (2) that $[\![p_-]\!]^{z \cdot vs}$ holds for any z, e.g. for $z < y$, where y is obtained from (16). Consequently z is a witness for p. Analogously we prove $\exists x.[\![p_+]\!]^{x \cdot vs} \to \exists x.[\![p]\!]^{x \cdot vs}$. This finishes the proof of $B \vee C \vee D \to A$. Now we only have to prove $A \wedge \neg B \wedge \neg C \to D$. For this assume $[\![p]\!]^{x \cdot vs}$ for some x and $\neg[\![p_-]\!]^{x \cdot vs}$ and $\neg[\![p_+]\!]^{x \cdot vs}$. This means that x is a withness for p that is neither too large nor too small. Hence x must lie in an interval with endpoints in $M_p = \{\frac{[\![t]\!]_\rho^{x \cdot vs}}{\underline{i}} | (t,i) \in \{\!\{\mathcal{U}\ p\}\!\}\}$. This is expressed by (18).

$$\begin{aligned} &\mathsf{islin}_{\mathcal{R}}\ p \wedge \neg[\![p_-]\!]^{x \cdot vs} \wedge \neg[\![p_+]\!]^{x \cdot vs} \wedge [\![p]\!]^{x \cdot vs} \\ &\to \exists((t,i),(s,j)) \in \{\!\{\mathcal{U}\ p\}\!\}^2. \frac{[\![t]\!]_\rho^{x \cdot vs}}{\underline{i}} \le x \le \frac{[\![s]\!]_\rho^{x \cdot vs}}{\underline{j}} \end{aligned} \tag{18}$$

The proof of (18) is easy. In fact its main part is done automatically. Now we conclude that either $x \in M_p$, in which case we are done (remember that $\frac{x+x}{2} = x$), or we can find the *smallest* interval with endpoints in M_p containing x, i.e. $l_x < x < u_x \wedge \forall y. l_x < y < u_x \to y \notin M_p$ for some $(l_x, u_x) \in M_p^2$. The construction of this *smallest* interval is simple.

Now we prove a main property of $\mathcal{R}$-formulae (19), which shows the the expressibility limitations of $\mathcal{R}$. A $\mathcal{R}$-formula p does *not* change its truth value over smallest intervals with endpoints in M_p, i.e.

$$\begin{aligned} &\mathsf{islin}_{\mathcal{R}}\ p \wedge l < x < u \wedge (\forall y. l < y < u \to y \notin M_p) \\ &\wedge [\![p]\!]^{x \cdot vs} \to \forall y. l < y < u \to [\![p]\!]^{y \cdot vs} \end{aligned} \tag{19}$$

The proof of (19) is by induction on p. The cases $\neq \mathbf{0}$ and $= \mathbf{0}$ are trivial. Assume p is $c * \boldsymbol{v}_0 + t < \mathbf{0}$ and let $z = -\frac{[\![t]\!]_\rho^{x \cdot vs}}{\underline{c}}$. From $[\![p]\!]^{x \cdot vs}$ we have $x < z$. Since $l < y < u$ and $z \in M_p$ we have $y \neq z$. Hence $y < z$ (which is $[\![p]\!]^{y \cdot vs}$), for if $y > z$ then $l < z < u$, which contradicts the premises since $z \in M_p$. The other interesting cases are proved analogously.

Since $[\![p]\!]^{x \cdot vs}$ and $l_x < x < u_x \wedge \forall y. l_x < y < u_x \to y \notin M_p$ for some $(l_x, u_x) \in M_p^2$, we conclude that $[\![p]\!]^{z \cdot vs}$ for any z such that $l_x < z < u_x$. Taking $z = \frac{l_x + u_x}{2}$ finishes the proof of (15).

In order to provide an implementation of ferrack, we define in Fig. 3 a function to simulate the substitution of $(\frac{[\![t]\!]_\rho^{x \cdot vs}}{\underline{i}} + \frac{[\![s]\!]_\rho^{x \cdot vs}}{\underline{j}})/\underline{2}$ for $\boldsymbol{v}_0$ in p, since division is not included in our language. We use the notation $p[(\frac{t}{\underline{i}} + \frac{s}{\underline{j}})/\underline{2}]$ for this substitution. The main property is expressed by

$$\begin{aligned} &\mathsf{islin}_{\mathcal{R}}\ p \wedge i > 0 \wedge j > 0 \wedge \mathsf{unbound}_\rho\ t \wedge \mathsf{unbound}_\rho\ s \\ &\qquad \to \mathsf{unbound}_\phi(p[(\frac{t}{\underline{i}} + \frac{s}{\underline{j}})/\underline{2}]) \\ &\wedge ([\![p[(\frac{t}{\underline{i}} + \frac{s}{\underline{j}})/\underline{2}]]\!]^{x \cdot vs} \leftrightarrow [\![p]\!]^{(([\![t]\!]_\rho^{x \cdot vs}/\underline{i} + [\![s]\!]_\rho^{x \cdot vs}/\underline{j})/\underline{2}) \cdot vs}) \end{aligned} \tag{20}$$

For the implementation of the bounded existential quantifier in (15) we use a function $\mathsf{eval}_\vee$, which basically evaluates a function f lazily over a list $[a_0, \ldots, a_n]$. The result represents $f\ a_0 \vee \ldots \vee f\ a_n$, i.e.

$$\forall vs, ps. [\![\mathsf{eval}_\vee\ f\ ps]\!]^{vs} \leftrightarrow \exists p \in \{\!\{ps\}\!\}. [\![f\ p]\!]^{vs} \tag{21}$$

$$\begin{aligned}
&(p \wedge q)[(\tfrac{t}{\underline{i}} + \tfrac{s}{\underline{j}})/\underline{2}] &&= p[(\tfrac{t}{\underline{i}} + \tfrac{s}{\underline{j}})/\underline{2}] \wedge q[(\tfrac{t}{\underline{i}} + \tfrac{s}{\underline{j}})/\underline{2}]\\
&(p \vee q)[(\tfrac{t}{\underline{i}} + \tfrac{s}{\underline{j}})/\underline{2}] &&= p[(\tfrac{t}{\underline{i}} + \tfrac{s}{\underline{j}})/\underline{2}] \vee q[(\tfrac{t}{\underline{i}} + \tfrac{s}{\underline{j}})/\underline{2}]\\
&(c * \boldsymbol{v}_0 + t' \ \bowtie \mathbf{0})[(\tfrac{t}{\underline{i}} + \tfrac{s}{\underline{j}})/\underline{2}] &&= 2{\cdot}j * t + 2{\cdot}i * s + 2{\cdot}i{\cdot}j * t' \ \bowtie \mathbf{0}\\
&p[(\tfrac{t}{\underline{i}} + \tfrac{s}{\underline{j}})/\underline{2}] &&= p\\
&\quad \mathsf{ferrack}\ p = \mathbf{let}\ \sigma = \lambda((t,i),(s,j)).p[(\tfrac{t}{\underline{i}} + \tfrac{s}{\underline{j}})/\underline{2}];\ U = \mathcal{U}\ p\\
&\qquad\qquad \mathbf{in}\ \mathsf{decr}(\mathsf{eval}_\vee\ \sigma\ (\mathsf{allpairs}\ U\ U))
\end{aligned}$$

Fig. 3. Substitution,$\mathsf{eval}_\vee$ and ferrack

The implementation of ferrack is given in Fig. 3. The function allpairs satisfies $\{\!\{\mathsf{allpairs}\ xs\ ys\}\!\} = \{\!\{xs\}\!\} \times \{\!\{ys\}\!\}$. For a $\mathcal{R}$-linear formula p, $[\![\mathsf{ferrack}\ p]\!]^{vs}$ is hence equivalent to

$$\exists((t,i),(s,j)) \in \{\!\{\mathcal{U}\ p\}\!\}^2. [\![p]\!]^{(([\![t]\!]_\rho^{x \cdot vs}/\underline{i} + [\![s]\!]_\rho^{x \cdot vs}/\underline{j})/\underline{2}) \cdot vs}.$$

The proof of (14) needs the following observation. Recall that the input to $qe_{\mathcal{R}_l}$ in mir (cf. § 3) is $p = \widehat{0} \leq \boldsymbol{v}_0 < \widehat{1} \wedge p'$, for some linear formula p' and hence $[\![p_-]\!]^{x \cdot vs} \leftrightarrow [\![p_+]\!]^{x \cdot vs} \leftrightarrow False$ and consequently ferrack correctly ignores p_- and p_+ (recall (15)). An implementation that covers all $\mathcal{R}$-linear formulae should simply include p_- and p_+.

5 Quantifier Elimination for $\mathcal{Z}$

We present cooper, a verified qep. for $\mathcal{Z}$ based on [12], and prove (22).

$$\mathsf{islin}_\mathcal{Z}\ p\ (\underline{i} \cdot vs) \rightarrow [\![\mathsf{cooper}\ p]\!]^{vs} \leftrightarrow \exists i. [\![p]\!]^{\underline{i} \cdot vs} \tag{22}$$

The input to Cooper's algorithm is a $\mathcal{Z}$-linear formula p. We only consider linear formulae where the coefficients of $\boldsymbol{v}_0$ are $\widehat{1}$, since $\exists i. Q(d{\cdot}i) \leftrightarrow \exists i. d \mid i \wedge Q(i)$ holds. It is straightforward to convert p into $p' = \mathsf{adjust}\ p\ d$, and prove (23), cf. [11].

$$\begin{aligned}
\mathsf{islin}_\mathcal{Z}\ p\ (\underline{i} \cdot vs) \wedge \mathsf{dvd}_{\widehat{c}}\ p\ d \wedge d > 0 \rightarrow \mathsf{islin}_\mathcal{Z}\ (\mathsf{adjust}\ p\ d)\ (\underline{i} \cdot vs)&\\
\wedge \mathsf{dvd}_{\widehat{c}}\ (\mathsf{adjust}\ p\ d)\ 1 \wedge [\![\mathsf{adjust}\ p\ d]\!]^{(\underline{d \cdot i}) \cdot vs} \leftrightarrow [\![p]\!]^{\underline{i} \cdot vs}& \qquad (23)\\
\mathsf{islin}_\mathcal{Z}\ p\ (\underline{i} \cdot vs) \rightarrow \mathsf{dvd}_{\widehat{c}}\ p\ (\mathsf{lcm}_{\widehat{c}}\ p) \wedge \mathsf{lcm}_{\widehat{c}}\ p > 0& \qquad (24)
\end{aligned}$$

The predicate $\mathsf{dvd}_{\widehat{c}}\ p\ d$ is true exactly when all the coefficients $\widehat{c}$ of $\boldsymbol{v}_0$ in p satisfy $c \mid d$. A candidate for d is $lcm\{c | c * \boldsymbol{v}_0$ occurs in $p\}$, which is computed recursively by $\mathsf{lcm}_{\widehat{c}}$, cf. (24).

$$\begin{aligned}\mathsf{cooper}\ p = \mathbf{let}\ & d = \mathsf{lcm}_{\widehat{c}}\ p; q = d \mid 1 * \boldsymbol{v}_0 + \widehat{0} \wedge (\mathsf{adjust}\ p\ d); \delta = \delta_q;\\ & M = \mathsf{eval}_{\vee}\ (\lambda j. q_-[\widehat{j}])[1..\delta];\\ & B = \mathsf{eval}_{\vee}\ (\lambda (b,j). q[b + \widehat{j}])\ (\mathsf{allpairs}\ (\mathcal{B}\ q)\ [1..\delta])\ \mathbf{in}\\ & \mathsf{decr}(M \vee B)\end{aligned}$$

Fig. 4. cooper

A fundamental property is that, for any $\mathcal{Z}$-linear p, the set $\{i | [\![p]\!]^{\underline{i} \cdot vs}\}$ differs from a periodic subset of $\mathbb{Z}$ *only* by a finite set (involving $\mathcal{B}\ p$, cf. Fig. 2). Let δ_p be $lcm\{d | d \mid 1 * \boldsymbol{v}_0 + t$ occurs in $p\}$, then (25) (Cooper's theorem [12]) expresses this fundamental property.

$$\begin{gathered}\mathsf{islin}_{\mathcal{Z}}\ p\ (\underline{i} \cdot vs) \wedge \mathsf{dvd}_{\widehat{c}}\ p\ 1 \rightarrow\\ \exists i. [\![p]\!]^{\underline{i} \cdot vs} \leftrightarrow \exists j \in \{1..\delta_p\}. [\![p_-]\!]^{\underline{j} \cdot vs} \vee \exists b \in \{\!\{\mathcal{B}\ p\}\!\}. [\![p]\!]^{(\underline{j} + [\![b]\!]_\rho^{\underline{i} \cdot vs}) \cdot vs}\end{gathered} \tag{25}$$

The proof is simple and we refer the reader to [12,28,10,20] for the mathematical details and to [11] for a verified formalization.

The implementation of cooper is shown in Fig. 4. First the coefficients of $\boldsymbol{v}_0$ are normalized to one. This step is correct by (23) and (24). After computing $\delta\ p$ and $\mathcal{B}\ p$, the appropriate disjunction is generated using $\mathsf{eval}_{\vee}$. The properties (21),(25) and (2) finish the proof of (22).

6 Formalization and Integration Issues

6.1 Normal Forms

When defining a function (over ρ or ϕ) we assume the input to have a precise syntactical shape, i.e. satisfy a given predicate. This not only simplifies the function definition but is also crucial for its correctness proofs. In [11], such functions used deeply nested pattern matching, which gives rise to a considerable number of equations, for the recursive definitions package avoids overlapping equations by performing completion. To avoid this problem, the ρ-datatype contains additional constructor $\boldsymbol{CX}\ int\ \rho$, not shown so far. Its intended meaning is $[\![\boldsymbol{CX}\ c\ t]\!]_\rho^{vs} = [\![c * \boldsymbol{v}_0 + t]\!]_\rho^{vs}$. In fact all the previous occurrences of $c * \boldsymbol{v}_0 + t$ can be understood as a syntactic sugar for $\boldsymbol{CX}\ c\ t$. Both proofs and implementation are simpler. In the ρ-datatype definition we also included only multiplication by a constant and it was maladroit not to do so in [11].

6.2 Optimizations

Our implementation includes not only the optimization presented in § 4, i.e. omitting the generation of p_- and p_+ in ferrack, but also several others, left out for space limitations. For instance several procedures scrutinize and simplify ρ-terms and ϕ-formulae. These are also used to keep the $\mathcal{U}$, cf. § 4, and $\mathcal{B}$, cf. § 5, small, which considerably affects the output size of ferrack and cooper. The evaluation of large disjunctions is done lazily. In [12], Cooper proved a dual lemma to (25) that uses substitution of arbitrary large numbers, cf. p_+ in Fig. 2 and a set $\mathcal{A}$, dual to $\mathcal{B}$. We formalized this duality principle, cf. [11] for more details, and choose the smaller set in the implementation. The generic qep. qelim_ϕ, cf. § 2.2, pushes $\exists$ inwards before every elimination. *All these optimizations are formally proved correct.*

6.3 Integration

We integrate the formalized qep. by providing an ML-function *reif*. Given a HOL subgoal P, it constructs a ϕ-formula p and a HOL-list vs such that the theorem $[\![p]\!]^{vs} = P$ can be proved in HOL. Obviously we can replace $[\![p]\!]^{vs}$ by $[\![\mathsf{qelim}_\phi\ \mathsf{mir}\, p]\!]^{vs}$ and then we either use rewriting or run the generated code, depending on our trust in the code generator. Of course *reif* can not succeed on every subgoal, since ϕ-formulae represent only a (small) subset of HOL-formulae. Note that the completeness of the integrated qep. relies *entirely* on the completeness of *reif*.

6.4 Performance

Since our development is novice, we have only tested the qep.for small reasonable looking examples. The generated code proves e.g. $\forall x.2{\cdot}\lfloor x\rfloor \leq \lfloor 2{\cdot}x\rfloor \leq 2{\cdot}\lfloor x+1\rfloor$ within 0.06 sec. but needs more than 10 sec. to prove $\forall x, y.\lfloor x\rfloor = \lfloor y\rfloor \to \underline{0} \leq |y-x| \leq \underline{1}$. The main causes are as follows:

– mir reduces the problem blindly to $\mathcal{R}$ and $\mathcal{Z}$, while often only $qe_{\mathcal{R}}$ or $qe_{\mathcal{Z}}$ is sufficient to eliminate $\exists$.
– The substitution in Fig. 3 gratuitously introduces big coefficients, which *heavily* influences the output size of cooper.
– cooper introduces big coefficients (which appear in $\cdot \mid \cdot$ atoms!), due to the global nature of the method (see [33]), which *heavily* influences the output size of $\mathsf{lin}_{\mathcal{R}}$.

Solving these problems is part of our future work.

7 Conclusion

We presented a formally verified and executable qep. for $\mathcal{R}_{\lfloor\cdot\rfloor}$, based on [38], and corroborate the maturity of modern theorem provers to assist formalizing state of the art qep. within acceptable time (1 month) and space (4000 lines). Our formalization includes a qep. for $\mathcal{R}$ à la [15] and Copper's qep. for $\mathcal{Z}$, that could be replaced by more efficient yet verified ones, e.g. [24,33]. Our work represents a new substantial application of reflection as well as a challenge for code generators, e.g. [7], to generate proof-producing code. Decision procedures developed this way are easier to maintain and to share with other theorem provers. This is one key issue to deal with the growing challenges, such as Flyspeck[1], modern theorem provers have to face.

Acknowledgment. I am thankful to Tobias Nipkow for suggesting the topic and for advice. I am also thankful to Clemens Ballarin, Michael Norrish and Norbert Schirmer for useful comments on a draft.

[1] `http://www.math.pitt.edu/~thales/flyspeck/`

References

1. Andrew W. Appel and Amy P. Felty. Dependent types ensure partial correctness of theorem provers. *J. Funct. Program.*, 14(1):3–19, 2004.
2. Henk Barendregt. Reflection and its use: from science to meditation, 2002.
3. Henk Barendregt and Erik Barendsen. Autarkic computations in formal proofs. *J. Autom. Reasoning*, 28(3):321–336, 2002.
4. Bruno Barras. Programming and computing in HOL. In *Proceedings of the 13th International Conference on Theorem Proving in Higher Order Logics*, pages 17–37. Springer-Verlag, 2000.
5. Sergey Berezin, Vijay Ganesh, and David L. Dill. An online proof-producing decision procedure for mixed-integer linear arithmetic. In Hubert Garavel and John Hatcliff, editors, *TACAS*, volume 2619 of *LNCS*, pages 521–536. Springer, 2003.
6. Stefan Berghofer. Towards generating proof producing code from HOL definitions. Private communication.
7. Stefan Berghofer and Tobias Nipkow. Executing higher order logic. In *In Types for Proofs and Programs (TYPES 2000)*, volume 2277 of *LNCS*, pages 24–40. Springer-Verlag, 2002.
8. Y. Bertot and P. Castéran. *Coq'Art: The Calculus of Inductive Constructions*, volume XXV of *Text in theor. comp. science: an EATCS series*. Springer, 2004.
9. Bernard Boigelot, Sébastien Jodogne, and Pierre Wolper. An effective decision procedure for linear arithmetic over the integers and reals. *ACM Trans. Comput. Log.*, 6(3):614–633, 2005.
10. A. Chaieb and T. Nipkow. Generic proof synthesis for presburger arithmetic. Technical report, Technische Universität München, 2003.
11. A. Chaieb and T. Nipkow. Verifying and reflecting quantifier elimination for Presburger arithmetic. In G. Stutcliffe and A. Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning*, volume 3835. Springer-Verlag, 2005.
12. D.C. Cooper. Theorem proving in arithmetic without multiplication. In B. Meltzer and D. Michie, editors, *Machine Intelligence*, volume 7, pages 91–100. Edinburgh University Press, 1972.
13. Pierre Crégut. Une procédure de décision réflexive pour un fragment de l'arithmétique de Presburger. In *Informal proceedings of the 15th journées francophones des langages applicatifs*, 2004. In French.
14. M. Davis. A computer program for presburger's algorithm. In *Summaries of talks presented at the Summer Inst. for Symbolic Logic, Cornell University*, pages 215–233. Inst. for Defense Analyses, Princeton, NJ, 1957.
15. Jeanne Ferrante and Charles Rackoff. A decision procedure for the first order theory of real addition with order. *SIAM J. Comput.*, 4(1):69–76, 1975.
16. Jeanne Ferrante and Charles Rackoff. *The Computational Complexity of Logical Theories*, volume 718 of *Lecture Notes in Mathematics*. Springer Verlag, NY, 1979.
17. Fischer and Rabin. Super-exponential complexity of presburger arithmetic. In *SIAMAMS: Complexity of Computation: Proc. of a Symp. in Appl. Math. of the AMS and the Society for Industrial and Applied Mathematics*, 1974.
18. J. Fourier. Solution d'une question particulière du calcul des inegalités. *Nouveau Bulletin des Sciences par la Scociété Philomatique de Paris*, pages 99–100, 1823.
19. John Harrison. Metatheory and reflection in theorem proving: A survey and critique. Technical Report CRC-053, SRI Cambridge, Millers Yard, Cambridge, UK, 1995. `http://www.cl.cam.ac.uk/users/jrh/papers/reflect.dvi.gz`.
20. John. R. Harrison. Introduction to logic and theorem proving. To appear.

21. Douglas J. Howe. Computational Metatheory in Nuprl. In Ewing L. Lusk and Ross A. Overbeek, editors, *CADE*, volume 310 of *LNCS*, pages 238–257, 1988.
22. Felix Klaedtke. On the automata size for Presburger arithmetic. In *Proceedings of the 19th Annual IEEE Symposium on Logic in Computer Science (LICS 2004)*, pages 110–119. IEEE Computer Society Press, 2004.
23. Robert Klapper and Aaron Stump. Validated Proof-Producing Decision Procedures. In C. Tinelli and S. Ranise, editors, *2nd Int. Workshop Pragmatics of Decision Procedures in Automated Reasoning*, 2004.
24. Rüdiger Loos and Volker Weispfenning. Applying linear quantifier elimination. *Comput. J.*, 36(5):450–462, 1993.
25. Sean McLaughlin and John Harrison. A proof-producing decision procedure for real arithmetic. volume 3632 of *LNCS*, pages 295–314. Springer-Verlag, 2005.
26. Sean McLauglin. An Interpretation of Isabelle/HOL in HOL Light. In U. Furbach and N. Shankar, editors, *Automated Reasoning — IJCAR 2006*, 2006. To appear.
27. Tobias Nipkow, Lawrence Paulson, and Markus Wenzel. *Isabelle/HOL — A Proof Assistant for Higher-Order Logic*, volume 2283 of *LNCS*. Springer-Verlag, 2002. `http://www.in.tum.de/~nipkow/LNCS2283/`.
28. Michael Norrish. Complete integer decision procedures as derived rules in HOL. In D.A. Basin and B. Wolff, editors, *Theorem Proving in Higher Order Logics, TPHOLs 2003*, volume 2758 of *LNCS*, pages 71–86. Springer-Verlag, 2003.
29. S. Obua and S. Skalberg. Importing HOL into Isabelle/HOL. In U. Furbach and N. Shankar, editors, *Automated Reasoning — IJCAR 2006*, 2006. To appear.
30. Derek C. Oppen. Elementary bounds for presburger arithmetic. In *STOC '73: Proceedings of the fifth annual ACM symposium on Theory of computing*, pages 34–37, New York, NY, USA, 1973. ACM Press.
31. Mojzesz Presburger. Über die Vollständigkeit eines gewissen Systems der Arithmetik ganzer Zahlen, in welchem die Addition als einzige Operation hervortritt. In *Comptes Rendus du I Congrès des Math. des Pays Slaves*, pages 92–101, 1929.
32. William Pugh. The Omega test: a fast and practical integer programming algorithm for dependence analysis. In *Proceedings of the 1991 ACM/IEEE conference on Supercomputing*, pages 4–13. ACM Press, 1991.
33. C. R. Reddy and D. W. Loveland. Presburger arithmetic with bounded quantifier alternation. In *STOC '78: Proceedings of the tenth annual ACM symposium on Theory of computing*, pages 320–325, New York, NY, USA, 1978. ACM Press.
34. T. Skolem. Über einige Satzfunktionen in der Arithmetik. In *Skrifter utgitt av Det Norske Videnskaps-Akademi i Oslo, I. Matematisk naturvidenskapelig klasse*, volume 7, pages 1–28. Oslo, 1931.
35. A. Tarski. *A Decision Method for Elementary Algebra and Geometry*. University of California Press, 2d edition, 1951.
36. Volker Weispfenning. The complexity of linear problems in fields. *J. Symb. Comput.*, 5(1/2):3–27, 1988.
37. Volker Weispfenning. The complexity of almost linear diophantine problems. *J. Symb. Comput.*, 10(5):395–404, 1990.
38. Volker Weispfenning. Mixed real-integer linear quantifier elimination. In *ISSAC '99: Proceedings of the 1999 international symposium on Symbolic and algebraic computation*, pages 129–136, New York, NY, USA, 1999. ACM Press.
39. Pierre Wolper and Bernard Boigelot. An automata-theoretic approach to presburger arithmetic constraints (extended abstract). In *SAS '95: Proc. of the Second Int. Symp. on Static Analysis*, pages 21–32, London, UK, 1995. Springer-Verlag.