CVC3 Proof Conversion to LFSC

Andrew Reynolds
Cesare Tinelli
Aaron Stump
The University of Iowa

Liana Hadarean
Yeting Ge
Clark Barrett
New York University

1 Introduction

This technical report gives definitions for conversion methods for proofs generated by the SMT solver Cvc3, into a format readable by the proof checker LFSC. We will discuss proofs in the quantifier-free linear real arithmetic logic (QF_LRA) of SMT.

LFSC ("Logical Framework with Side Conditions") is a proof checker based on the Edinburgh Logical Framework (LF), a high-level declarative language in which logics (understood as inference systems over a certain language of formulas) can be specified. LFSC increases LF's flexibility by including support for computational side conditions on inference rules. These conditions, expressed in a small functional programming language, enable some parts of a proof to be established by computation.

In this work, proofs in the LFSC calculus were translated from proofs produced by Cvc3 in its own calculus. Since Cvc3's proof-generation facility is deeply embedded in the system's code, a translation module was added to Cvc3 that traverses the internal data structure storing the proof, and produces an LFSC proof from it. This translation module consisted of three translation strategies (which we will call LRA1, LRA2, LRA2a), varying in the degree of computational side conditions in which they incorporate.

The core of this document will be devoted to a formal definition of these translations.

Document outline. Section 2 introduces necessary definitions, including QF_LRA terminology and a definition of a proof datatype. Section 3 describes a high level view of the three translation methods from Cvc3 to LFSC proofs. Section 4 gives an overview of the proof calculi on which these translations operate.

A technical description of three Cvc3 to LFSC proof translations is provided in Sections 5-7. Section 5 describes the LRA1 translation that remains mostly faithful to the structure of the original Cvc3 proof. Section 6 and 7 describe two alternate translations (LRA2 and LRA2a) that attempt to compact portions of the Cvc3 proof into computational side conditions.

Section 8 details the compression we achieve when converting to proof rules in a proof calculus involving computational side conditions.

2 Preliminaries

2.1 LRA Terms

Define rational constants c and terms t to be of the following format:

$$
\begin{array}{l}
c ::= n_1 \mid \frac{n_1}{n_2} \\
t ::= c \mid v \mid t_1 + t_2 \mid t_1 - t_2 \mid c \cdot t_1 \mid t_1 \cdot c \mid ite(\varphi, t_1, t_2)
\end{array}
$$

where n_1 is an integer numeral, n_2 is a non-zero integer numeral, and v is a

2.2 QF_LRA Formulas

The following is a describes the format for all QF_LRA formulas used by Cvc3. We will refer to formulas φ^a as *theory atoms*.

$$
\begin{array}{l}
\varphi^a ::= t_1 = t_2 \mid t_1 > t_2 \mid t_1 \geq t_2 \mid t_1 \neq t_2 \mid t_1 < t_2 \mid t_1 \leq t_2 \\
\varphi ::= \varphi^a \mid \bot \mid \top \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \Rightarrow \varphi_2 \mid \varphi_1 \Leftrightarrow \varphi_2 \mid ite(\varphi, \varphi_1, \varphi_2)
\end{array}
$$

We will write $\sim$ to denote an element of $\{=, >, \geq, \neq, <, \leq\}$, $\succ$ will denote an element of $\{\geq, >\}$, and $\prec$ will denote an element of $\{\leq, <\}$. When $\sim$ is $=, >, \geq, \neq, <, \leq$, we will write $\nsim$ to denote $\neq, \leq, <, =, \geq, >$ respectively, and write $\sim_\downarrow$ to denote $=, >, \geq, \neq, >, \geq$ respectively. We will also write $(\sim_1 \cdot \sim_2)$ to denote the resulting relation according to the *lra_add* rules (or their unnormalized equivalents) in the $\mathcal{L}$ calculus for $\sim_1$ and $\sim_2$. For example $(> \cdot >)$ is $>$ and $(= \cdot \geq)$ is $\geq$.

2.3 Proofs

Formally introduce a proof datatype P as a triple, containing a set of subproofs, rule instance, and conclusion formula. We say that $P : \Gamma \vdash \varphi$ iff
(1) P is one of:
 $(\{\}$, **assert** , $\varphi)$, where $\varphi \in \Gamma$, or
 $(\{P_1, \ldots P_n\}$, r, $\varphi)$
(2) $P_i : \Gamma_i \vdash \varphi_i$ for all i, for some φ_i and Γ_i,
(3) applying r to $\varphi_1 \ldots \varphi_n$ produces φ

For a proof P to be well-formed, we also require a legal choice of Γ according to what is specified by rule r. We will write proofs P graphically as the following, where $P_1 \ldots P_n$ are the proofs of the premises of P:

$$
\frac{P_1 : \Gamma_1 \vdash \varphi_1 \quad \ldots \quad P_n : \Gamma_n \vdash \varphi_n}{P : \Gamma \vdash \varphi} \ \mathsf{r}
$$

We will omit annotations (P :) for unnamed subproofs and write $P : \varphi$ as shorthand for $P : \Gamma \vdash \varphi$ when Γ is understood or is not important.

2.4 Polynomials

In order to efficiently manipulate linear real arithmetic terms, LFSC will operate terms that are normalized to a linear *polynomial* form. A linear polynomial is of the form $(c_1 \cdot v_1 + \ldots + c_n \cdot v_n) + c$, where each c_i is a rational constant, each v_i is a real variable. We will write the symbol p (possibly with subscripts) to denote such polynomials. Furthermore, we will refer to *polynomial atoms* of the form $p \sim 0$, denoting a formula whose left hand side is an instance of a polynomial.

We will write $e\!\downarrow$ to denote the result of normalizing the expression e to a polynomial. In the case of normalization occurring in the conclusion of a proof rule, this normalization is done by the rule's side condition, which is left implicit to keep the notation uncluttered.

3 Proof Generation

Proofs in our LFSC calculus for LRA are generated from proofs produced by Cvc3 in its own calculus. We will refer to the former calculus as $\mathcal{L}$ and the latter as $\mathcal{C}$.

Cvc3 Proof structure Roughly speaking, Cvc3's proofs have a two-tiered structure, typical of solvers based on the DPLL(T) architecture [?], with a propositional skeleton filled with several theory-specific subproofs. The conclusion is reached by means of propositional or purely equational inferences applied to a set of input formula and a set of *theory lemmas*. The latter are disjunctions of arithmetic atoms deduced from no assumptions, mostly using proof rules specific to the theory in question—the theory of real arithmetic in this case.

In order to experiment with the declarative/computational continuum, we implemented three different translations from Cvc3 proofs, differing in how close they are to the original proof. We refer to these as the *literal*, the *liberal* and the *aggressively liberal* translation, and name them LRA1, LRA2, and LRA2a, respectively.

Literal translation. In the literal translation, LRA1, an LFSC proof is produced directly from Cvc3's proof, using whenever possible $\mathcal{L}$ rules that mirror the corresponding $\mathcal{C}$ rules, and resorting to additional $\mathcal{L}$-specific rules only for those few $\mathcal{C}$ rules that cannot be checked by simple pattern matching (but require, for instance, to verify that a certain expression in the $\mathcal{C}$ rule is a normalized version of another).

Liberal translation. In the liberal translation LRA2, the Cvc3 proof is used as a guide to produce a compact proof that relies on rules with side conditions specific to $\mathcal{L}$—that is, not encoding a rule of $\mathcal{C}$. The use of side conditions enables compaction that is otherwise infeasible due to the declarative nature of rules in the $\mathcal{C}$ calculus. In LRA2, the subproofs of all theory lemmas are systematically converted to more compact proofs that use $\mathcal{L}$-specific rules; the rest of the Cvc3 proof is translated as in the literal translation.

$$\frac{\varphi_1 \Leftrightarrow \varphi_2 \quad \varphi_2 \Leftrightarrow \varphi_3}{\varphi_1 \Leftrightarrow \varphi_3}\ \textsf{iff\_trans} \qquad \frac{\varphi_1 \quad \varphi_1 \Leftrightarrow \varphi_2}{\varphi_2}\ \textsf{iff\_mp}$$

$$\frac{t_1 = t_2 \quad t_3 = t_4}{t_1 \sim t_3 \Leftrightarrow t_2 \sim t_4}\ \textsf{congr\_1} \qquad \frac{t_1 = t_2 \quad t_2 = t_3}{t_1 \ = t_3}\ \textsf{eq\_trans}$$

$$\frac{t_1 = t_2 \quad t_3 = t_4}{t_1 \bowtie t_3 = t_2 \bowtie t_4}\ \textsf{congr\_2} \qquad \frac{t_1 = t_2}{t_2 = t_1}\ \textsf{eq\_symm}$$

$$\frac{t_1 > t_2 \quad t_2 > t_3}{t_1 > t_3}\ \textsf{gt\_trans} \qquad \frac{t_1 > t_2 \quad t_2 > t_1}{\bot}\ \textsf{gt\_acyc}$$

$$\frac{\{0 \not\sim c\}}{(0 \sim c) \Leftrightarrow \bot}\ \textsf{const\_pred\_1} \qquad \frac{}{t_1 \sim t_2 \Leftrightarrow 0 \sim t_2 - t_1}\ \textsf{right\_minus\_left}$$

$$\frac{\{t' \text{ canonical form of } t\}}{t = t'}\ \textsf{canon} \qquad \frac{\{c \text{ non-negative}\}}{t_1 \sim t_2 \Leftrightarrow c \cdot t_1 \sim c \cdot t_2}\ \textsf{mult\_pred}$$

$$\frac{}{t_1 > t_2 \Leftrightarrow t_2 < t_1}\ \textsf{flip\_ineq} \qquad \frac{}{t_1 \sim t_2 \Leftrightarrow t_1 + t_3 \sim t_2 + t_3}\ \textsf{plus\_pred}$$

Figure 1: Some of Cvc3's proof rules for QF_LRA.

Aggressively Liberal translation. The LRA2a translation is identical to LRA2 except that it tries to compact also parts of the proof that rely on generic equality reasoning (for instance, applications of congruence rules), again by using $\mathcal{L}$-specific rules. This translation uses an adaptive strategy to switch from $\mathcal{L}$-specific equality rules to $\mathcal{C}$-like equality rules and back, making heuristic decisions on when it is worthwhile to do so. We will see that this switching requires some additional overhead.

4 The Cvc3 and LFSC Calculi for LRA

4.1 The $\mathcal{C}$ Calculus

Although implemented in Cvc3 as a sequent calculus, the fragment of Cvc3's proof system for QF_LRA can be described mathematically as a natural deduction calculus. A proof in the $\mathcal{C}$ calculus derives a quantifier-free formula φ from a set of assumed LRA formulas Γ, all of which are also quantifier-free.

A sample of Cvc3's rules is provided in Figure 1.[1] Most of the rules are fairly standard and self-explanatory, with the possible exception of canon, which asserts an equality between a term t and its equivalent *canonical form* produced by Cvc3's canonizer module. As a whole, these rules are used to represent a trace of the reasoning used by Cvc3's decision procedure for QF_LRA.

Although the $\mathcal{C}$ calculus itself is quite general, all Cvc3 proofs in it are *refutations*, that is, they prove $\bot$ from a set of assumptions Γ, where Γ is a subset of the formulas whose joint satisfiability Cvc3 was asked to check.

[1] A more extensive set of rules is provided in the appendix. Note the complete proof system is a lot bigger because it supports a much larger logic than QF_LRA.

$$\frac{p = 0}{(c \cdot p)\!\downarrow = 0}\ \textsf{lra\_mult\_c=} \qquad \frac{p > 0 \quad \{c > 0\}}{(c \cdot p)\!\downarrow > 0}\ \textsf{lra\_mult\_c>}$$

$$\frac{p_1 = 0 \quad p_2 \sim 0}{(p_1 + p_2)\!\downarrow \sim 0}\ \textsf{lra\_add=}\!\sim \qquad \frac{p_1 \sim 0 \quad p_2 = 0}{(p_1 - p_2)\!\downarrow \sim 0}\ \textsf{lra\_sub}\!\sim\!\textsf{=}$$

$$\frac{\{c \sim 0\}}{c \sim 0}\ \textsf{lra\_axiom}\!\sim \qquad \frac{p \sim 0 \quad \{p \nsim 0\}}{\bot}\ \textsf{lra\_contra}\!\sim$$

$$\frac{p \geq 0 \quad p' \geq 0 \quad \{p + p' = 0\}}{p = 0}\ \textsf{lra}\!\geq\!\geq\!\textsf{to=}$$

Figure 2: Some of the polynomial rules of $\mathcal{L}$.

4.2 The $\mathcal{L}$ Calculus

The LFSC calculus for $\mathcal{L}$ can be described as a proper superset of $\mathcal{C}$. For the purposes of optimization, both liberal translations use rules to convert arithmetic terms—denoted by the letter t in the rules—to polynomials—denoted by p. [2]

A further set of rules operate only on polynomial atoms and are used by the liberal translations to generate proofs of LRA lemmas. A sample of these rules is provided in Figure 2. [3] To ease formatting, side conditions are written together with the premises, but enclosed in braces. Although side conditions use the same syntax used in the sequents, they should be read as a mathematical notation. For example, $p = 0$ in a premise denotes an atomic formula whose left-hand side is an arbitrary polynomial and whose right-hand side is the 0 polynomial; in contrast, the side condition $\{p + p' = 0\}$, say, denotes the result of checking whether the expression $p + p'$ evaluates to 0 in the polynomial ring $\mathbb{Q}[X]$, where $\mathbb{Q}$ is the field of rational numbers and X the set of all variables (or "free constants" in SMT-LIB parlance).

In the following sections, we use the terminology $\mathcal{C}$ *proof* to refer to a proof whose rule r belongs to the $\mathcal{C}$ calculus, and similarly for $\mathcal{L}$ proof. Note we do not impose restrictions on the types of subproofs in the proof datatype.

5 Literal Conversion

Translation T_{LRA1} Define a proof translation operator $T_{\mathsf{LRA1}} : \mathbf{P} \to \mathbf{P}$ from $\mathcal{C}$ proofs to $\mathcal{L}$ proofs. This translation is faithful, that is to say:

Lemma 1 If $P : \Gamma \vdash \varphi$, then $T_{\mathsf{LRA1}}(P) : \Gamma \vdash \varphi$.

Although faithful with respect to what is proven, in some cases, T_{LRA1} may change the concrete syntax of proofs, including rule names, as well as structural details that come as consequence of our LFSC implementation. In this section, we will discuss such exceptions for $T_{\mathsf{LRA1}}(P)$.

[2] These conversion rules can be found in Appendix B.7.

[3] A complete set of $\mathcal{L}$-specific rules is provided in the appendix.

5.1 Canon

Cvc3 proofs include a variety of canonize rules (including canon_mult, canon_plus, canon_invert_divide), all of which can be summarized by the **canon** rule. Say that P is a proof of the following form:

$$\frac{\{t' \text{ canonical form of } t\}}{P : t = t'} \ \textsf{canon}$$

Instead of explicitly modeling the Cvc3 canonizer in LFSC, we define $T_{\mathsf{LRA1}}(P)$ as the following:

$$\frac{P'_1 : t = p \quad P'_2 : t' = p' \quad \{(p - p')\downarrow = 0\}}{T_{\mathsf{LRA1}}(P) : t = t'} \ \textsf{canon}$$

This is to say, we first normalize both t and t' to their polynomial form using proofs P'_1 and P'_2, and use a computational side condition to verify that $p - p'$ normalizes to the constant polynomial 0.

5.2 rewrite_and, rewrite_or

Cvc3 uses the coarse grained rules rewrite_and, and rewrite_or to deal with associativity of conjunctions and disjunctions respectively. We will discuss rewrite_and in this section, noting that rewrite_or is translated analogously. Say P is a proof of the following form:

$$\frac{\{\varphi' \text{ canonical form of } \varphi\}}{P : \varphi \Leftrightarrow \varphi'} \ \textsf{rewrite\_and}$$

where φ is $\varphi_1 \wedge \ldots \wedge \varphi_n$ with an arbitrary parenthesization, and φ' is equivalent to φ with a different parenthesization, canonized according to Cvc3.

The corresponding rule in LFSC is very similar. Instead of modeling the Cvc3 canonizer, we use a side condition to compute φ'. We define $T_{\mathsf{LRA1}}(P)$ to be the following, where $\varphi\downarrow$ represents the result of reassociating φ according to the LFSC side condition "normalize_and":

$$\frac{\{\varphi \downarrow = \varphi'\}}{T_{\mathsf{LRA1}}(P) : \varphi \Leftrightarrow \varphi'} \ \textsf{rewrite\_and}$$

Since we were unable to fully simulate the canonize method used by Cvc3, there are cases in which we must accept an instance of this rule as an axiom.

5.3 cycleConflict

The coarse grained Cvc3 rule cycleConflict takes a variable number of premise inequalities which are jointly unsatisfiable. Say P is a proof of the following form:

$$\frac{P_1 : t_1 \prec_1 t'_1 \quad \ldots \quad P_n : t_n \prec_n t'_n}{\bot}\ \textsf{cycleConflict}$$

where $(t_1 \prec_1 t'_1) \wedge \ldots \wedge (t_n \prec_n t'_n) \Rightarrow \bot$. [4]

To prove these premises to be unsatisfiable, we will first normalize all premises to polynomial form (using a polynomial normalization operator T_p that will be defined in Section 6.1) and sum them to obtain an inconsistent polynomial equation.

Since the LFSC framework does not support type definitions taking a variable number of arguments, we cascade a chain of corresponding polynomial addition operations. We define $T_{\textsf{LRA1}}(P)$ as the following:

$$\cfrac{\cfrac{\cfrac{P'_1 : p_1 \succ_1 0 \quad P'_2 : p_2 \succ_2 0}{(p_1 + p_2){\downarrow} \succ'_2 0}\ \textsf{lra\_add}{\succ_1}{\succ_2} \qquad \vdots \qquad P'_n : p_n \succ_n 0}{(p_1 + \ldots + p_n){\downarrow} \succ'_{n-1} 0}\ \textsf{lra\_add}{\succ'_{n-1}}{\succ_n}}{T_{\textsf{LRA1}}(P) : \bot}\ \textsf{lra\_contra}{\succ'_n}$$

where $P'_i = T_p(T_{\textsf{LRA1}}(P_i))$, $\succ'_1$ is $\succ_1$, and $\succ'_i$ is $(\succ'_{i-1} \cdot \succ_i)$ for $i > 1$.

We claim that the resultant summation $(p_1 + \ldots + p_n){\downarrow} \succ'_{n-1} 0$ is indeed a contradiction. As a consequence of footnote 4, we have that $(p_1 + \ldots + p_n){\downarrow} = (t_2 - t_1) + (t_3 - t_2) + \ldots (t_1 - t_n)){\downarrow} = 0$, and $\succ'_{n-1}$ is $>$, giving us $0 > 0$, a contradiction.

Note that because cycleConflict is coarse grained, requires a fairly lengthy corresponding proof in LFSC. However, note that the overhead incurred for this rule by the literal translation will be comparable to the overhead incurred by the liberal translations LRA2 and LRA2a.

5.4 optimized_subst_op

= **Case** Say P is a proof of the following form:

$$\frac{P_1 : t_1 = s_1 \quad \ldots \quad P_n : t_n = s_n}{P : t = t[s_1/t_1 \ldots s_n/t_n]}\ \textsf{optimized\_subst\_op}_1$$

Here, proof P has n subproofs of equalities $t_i = s_i$. The rule optimized_subst_op$_1$ will replace s_i for t_i within the right hand side of the conclusion. The conclusion is described in general terms, although the location of replacements s_i/t_i is restricted and occurs in only one location per pair of terms. Additionally, t is either (1) an ite expression or (2) an expression of the form $t_1 \bowtie \ldots \bowtie t_n$, where $\bowtie \in \{+, -, \cdot\}$.

In the case when t is an ite expression, we make use of the following two rules:

[4] More precisely, we have that $t_1 \prec_1 t'_1 = t_2 \prec_2 t'_2 = \ldots = t_n \prec_n t'_n = t_1$, where at least one of $\succ_1 \ldots \prec_n$ is $<$.

$$\frac{ite(\psi_1, t', t_1) = ite(\psi_2, t', s_1) \quad t_2 = s_2}{ite(\psi_1, t_2, t_1) = ite(\psi_2, s_2, s_1)}\ \mathsf{ite\_t1}_{oso1}$$

$$\frac{ite(\psi_1, t_1, t') = ite(\psi_2, s_1, t') \quad t_2 = s_2}{ite(\psi_1, t_1, t_2) = ite(\psi_2, s_1, s_2)}\ \mathsf{ite\_t2}_{oso1}$$

We cascade (up to two) applications of these rules, and overall define $T_{\mathsf{LRA1}}(P)$ as the following:

$$\dfrac{\dfrac{\dfrac{}{ite(\psi, t, t') = ite(\psi, t, t')}\ \mathsf{refl} \quad T_{\mathsf{LRA1}}(P_1) : t_1 = s_1}{ite(\psi, t_1, t') = ite(\psi, s_1, t')}\ 1 \quad T_{\mathsf{LRA1}}(P_2) : t_2 = s_2}{T_{\mathsf{LRA1}}(P) : ite(\psi, t_1, t_2) = ite(\psi, s_1, s_2)}\ 2$$

where 1 is $\mathsf{ite\_t1}_{oso1}$ and 2 is $\mathsf{ite\_t2}_{oso1}$.

Otherwise, when t is an expression of the form $t_1 \bowtie \ldots \bowtie t_n$, we define $T_{\mathsf{LRA1}}(P)$ as the following:

$$\dfrac{\dfrac{\dfrac{T_{\mathsf{LRA1}}(P_1) : t_1 = s_1 \quad T_{\mathsf{LRA1}}(P_1) : t_2 = s_2}{t_1 \bowtie t_2 = s_1 \bowtie s_2}\ 1}{\vdots} \quad T_{\mathsf{LRA1}}(P_n) : t_n = s_n}{T_{\mathsf{LRA1}}(P) : t_1 \bowtie \ldots \bowtie t_n = s_1 \bowtie \ldots \bowtie s_n}\ 1$$

where 1 is basic_subst_op1.

⇔ Case Say P is a proof of the following form:

$$\frac{P_1 : \varphi_1 \Leftrightarrow \psi_1 \quad \ldots \quad P_n : \varphi_n \Leftrightarrow \psi_n}{P : \varphi \Leftrightarrow \varphi[\psi_1/\varphi_1, \ldots, \psi_n/\varphi_n]}\ \mathsf{optimized\_subst\_op}_2$$

Similarly to the previous section, φ is either (1) an ite expression, or (2) a formula of the form $\varphi_1 \Box \ldots \Box \varphi_n$ where $\Box \in \{\wedge, \vee, \Leftrightarrow\}$.

In the first case, when φ is an ite formula, we use the following two rules:

$$\frac{ite(\psi, \varphi', \varphi_1) \Leftrightarrow ite(\psi', \varphi', \psi_1) \quad \varphi_2 \Leftrightarrow \psi_2}{ite(\psi, \varphi_2, \varphi_1) \Leftrightarrow ite(\psi', \psi_2, \psi_1)}\ \mathsf{ite\_}\varphi\mathsf{1}_{oso1}$$

$$\frac{ite(\psi, \varphi_1, \varphi') \Leftrightarrow ite(\psi', \psi_1, \varphi') \quad \varphi_2 \Leftrightarrow \psi_2}{ite(\psi, \varphi_1, \varphi_2) \Leftrightarrow ite(\psi', \psi_1, \psi_2)}\ \mathsf{ite\_}\varphi\mathsf{2}_{oso1}$$

We cascade (up to two) applications of these rules, and overall define $T_{\mathsf{LRA1}}(P)$ as the following:

$$\dfrac{\dfrac{\dfrac{}{ite(\psi,\varphi,\varphi') \Leftrightarrow ite(\psi,\varphi,\varphi')}\ \textsf{refl} \quad T_{\textsf{LRA1}}(P_1) : \varphi_1 \Leftrightarrow \psi_1}{ite(\psi,\varphi_1,\varphi') \Leftrightarrow ite(\psi,\psi_1,\varphi')}\ \textsf{1} \quad T_{\textsf{LRA1}}(P_2) : \varphi_2 \Leftrightarrow \psi_2}{T_{\textsf{LRA1}}(P) : ite(\psi,\varphi_1,\varphi_2) \Leftrightarrow ite(\psi,\psi_1,\psi_2)}\ \textsf{2}$$

where 1 is $\textsf{ite\_}\varphi\textsf{1}_{oso1}$ and 2 is $\textsf{ite\_}\varphi\textsf{2}_{oso1}$.

Otherwise, when t is an expression of the form $\varphi_1 \Box \ldots \Box \varphi_n$, we will define $T_{\textsf{LRA1}}(P)$ as the following:

$$\dfrac{\dfrac{\dfrac{T_{\textsf{LRA1}}(P_1) : \varphi_1 \Leftrightarrow \psi_1 \quad T_{\textsf{LRA1}}(P_2) : \varphi_2 \Leftrightarrow \psi_2}{\varphi_1 \Box \varphi_2 \Leftrightarrow \psi_1 \Box \psi_2}\ \textsf{1}}{\vdots} \quad T_{\textsf{LRA1}}(P_n) : \varphi_n \Leftrightarrow \psi_n}{T_{\textsf{LRA1}}(P) : \varphi_1 \Box \ldots \Box \varphi_n \Leftrightarrow \psi_1 \Box \ldots \Box \psi_n}\ \textsf{1}$$

where 1 is $\textsf{basic\_subst\_op}_2$.

5.5 Default Case

In all other cases for which P is of the following form:

$$\dfrac{P_1 : \varphi_1 \quad \ldots \quad P_n : \varphi_n}{P : \varphi}\ \textsf{r}$$

We define $T_{\textsf{LRA1}}(P)$ as the proof:

$$\dfrac{T_{\textsf{LRA1}}(P_1) : \varphi_1 \quad \ldots \quad T_{\textsf{LRA1}}(P_n) : \varphi_n}{T_{\textsf{LRA1}}(P) : \varphi}\ \textsf{r'}$$

where r' is the corresponding rule name for r in the LFSC signature.

6 Aggressive Liberal Conversion

In this section we will define the aggressive liberal translation $T_{\textsf{LRA2a}}$. This translation will be defined in terms of four operators, the first being the literal translation $T_{\textsf{LRA1}}$ as defined in Section 5. The second, T_p will be our method of concluding normalized polynomial formulas from term formulas. The third, T will refer to proof compression technique involving proofs about normalized polynomials. The fourth, T_p^{-1} will be a method of constructing $T_{\textsf{LRA1}}$ proofs from polynomial proofs.

6.1 Polynomial Normalization Operator T_p

For each theory atom φ proven by the Cvc3 proof, we will associate a unique polynomial atom $p \sim 0$ such that $p \sim 0$ is logically equivalent to φ, that is, $p \sim 0$ is true in exactly the same valuations in which φ is. For example, for the equality atom $2x = 2y$, this polynomial is $(2x - 2y)\!\downarrow = 0$.

For a theory atom (or negation thereof) φ, we will denote its polynomial equivalent with the notation φ^p. This correspondence is defined as follows:

$$\begin{array}{lll} (t_1 \sim t_2)^p & := & (t_1 - t_2)\!\downarrow \sim 0 \quad \text{for} \sim \in \{=, \geq, >\} \\ (t_1 \sim t_2)^p & := & (t_2 - t_1)\!\downarrow \sim_\downarrow 0 \quad \text{for} \sim \in \{\neq, \leq, <\} \\ (\neg(t_1 \sim t_2))^p & := & (t_1 \nsim t_2)^p \\ (\neg\neg\varphi)^p & := & \varphi^p \end{array}$$

Lemma 2

If $\varphi^p ::= p \sim 0$, then $(\neg\varphi)^p ::= (-p)\!\downarrow(\nsim)_\downarrow 0$.

Lemma 3

$\varphi^p \leftrightarrow \varphi$.

We claim that by using the poly_norm proof rules used in the $\mathcal{L}$ calculus as well as rules for eliminating negations from theory literals, we can define a proof translation function $T_p : \mathbf{P} \rightarrow \mathbf{P}$ from $\mathcal{L}$ proofs to $\mathcal{L}$ proofs such that:

Lemma 4

If $P : \Gamma \vdash \varphi$ and φ^p is defined, then $T_p(P) : \Gamma \vdash \varphi^p$.

The precise defintion of T_p and proof of Lemma 4 is omitted here. The general idea is that we apply normalization inductively over the structure of terms (using rules in Section B.7) until we are able to apply a equation normalization rule (defined in Section B.8) to convert our statement involving terms to one involving polynomials.

6.2 Polynomial Operator T

Define a proof translation operator $T : \mathbf{P}_{lra} \rightarrow \mathbf{P}$ from theory reasoning $\mathcal{C}$ proofs [5] to $\mathcal{L}$ proofs. This translation is performed incrementally and bottom-up over the structure of the Cvc3 proof, where applications of rules in $\mathcal{C}$ are translated to applications of corresponding rules for polynomials in $\mathcal{L}$. The translation will rely on the following invariant:

Invariant 1

(a) For all theory reasoning $\mathcal{C}$ proofs $P : \Gamma \vdash \varphi$ where φ^p is defined:
- (i) $T(P) : \Gamma \vdash \varphi^p$.

(b) For all theory reasoning $\mathcal{C}$ proofs $P : \Gamma \vdash \varphi_1 \Leftrightarrow \varphi_2$, there is a constant c s.t.:
- (i) $T(P) : \Gamma \vdash (c \cdot p_1 - p_2)\!\downarrow = 0$,
- (ii) $\varphi_1^p ::= (p_1 \sim 0)$,
- (iii) $\varphi_2^p ::= (p_2 \sim 0)$,
- (iv) $c > 0$ (if $\sim$ is $>$ or $\geq$), $c \neq 0$ otherwise.

(c) For all theory reasoning $\mathcal{C}$ proofs $P : \Gamma \vdash \varphi \Leftrightarrow \top$ where φ^p is defined:
- (i) $T(P) : \Gamma \vdash \varphi^p$.

(d) For all theory reasoning $\mathcal{C}$ proofs $P : \Gamma \vdash \varphi \Leftrightarrow \bot$ where φ^p is defined:

[5] A proof $P : \varphi$ is a *theory reasoning $\mathcal{C}$ proof* if and only if Invariant 1 is defined for P, and all premise subproofs of P are also theory reasoning $\mathcal{C}$ proofs.

(i) $T(P) : \Gamma \vdash (\neg\varphi)^p$.

(e) For all theory reasoning $\mathcal{C}$ proofs $P : \Gamma \vdash \bot$:

(i) $T(P) : \Gamma \vdash \bot$.

Our definition of Invariant 1 is slightly simplified here for the purposes of clarity. There are specific instances in which our corresponding proof $T(P)$ may prove something strictly stronger than what is specified by Invariant 1. Such cases come as a consequence of a sixth case of Invariant 1 that is defined for proofs of the form $P : \Gamma \vdash \varphi_1 \Rightarrow \varphi_2$, which is not mentioned here.

Note that Invariant 1 covers all cases of our definition of T, due to the following lemma:

Lemma 5

If $T(P)$ is defined, then Invariant 1 holds for P.

All translated proofs $T(P)$ are at least as strong as the original proof P, due to the following theorem:

Theorem 1 If $P : \Gamma \vdash \varphi$ and $T(P) : \Gamma \vdash p \sim 0$, then $p \sim 0$ implies φ.

Proof The proof is constructive and follows from each case of Invariant 1. In Section 6.3, we will define a translation T_p^{-1}, such that whenever $P : \Gamma \vdash \varphi$ and $T(P) : \Gamma \vdash p \sim 0$, we have that $T_p^{-1}(T(P)) : \Gamma \vdash \varphi$. This will suffice as a proof of Theorem 1, under the assumption that our $\mathcal{L}$ calculus is sound. □

We will now give a formal definition of all theory reasoning proof rules handled by our translation function T, and show that Invariant 1 is locally maintained in each case.

6.2.1 Assertions

Say proof P is an assertion of the assumption φ, for some $\varphi \in \Gamma$. We claim that all Cvc3 assertions φ in the QF_LRA logic are such that φ^p is defined. In this case, we define $T(P) = T_p(P')$, where P' is the corresponding assertion of φ in the $\mathcal{L}$ calculus. Furthermore, since φ^p is defined, by Lemma 4, we have that $T(P) : \Gamma \vdash \varphi^p$, and thus Invariant 1(a) holds for P.

6.2.2 iff_trans

$\top$ **Case** Consider when P is a proof of the following form:

$$\frac{P_1 : \varphi_1 \Leftrightarrow \varphi_2 \quad P_2 : \varphi_2 \Leftrightarrow \top}{P : \varphi_1 \Leftrightarrow \top}\ \textsf{iff\_trans}$$

By assumption of Invariant 1(b) for P_1, we have that $T(P_1) : (c{\cdot}p_1 - p_2){\downarrow} = 0$, where $\varphi_1^p ::= (p_1 \sim 0)$ and $\varphi_2^p ::= (p_2 \sim 0)$. Note that property (iv) of Invariant 1(b) guarantees that a multiplication of an equation of the form $p \sim 0$ by the constant $\frac{1}{c}$ is legal. Assume Invariant 1(c) holds for P_2. The following defines the transformed proof $T(P)$:

$$\cfrac{\cfrac{T(P_1) : (c \cdot p_1 - p_2)\!\downarrow = 0 \quad T(P_2) : (p_2 \sim 0)}{(c \cdot p_1 - p_2 + p_2)\!\downarrow \sim 0}\ \textsf{lra\_add=}\!\sim}{(\frac{1}{c} \cdot (c \cdot p_1 - p_2 + p_2))\!\downarrow \sim 0}\ \textsf{lra\_mult\_c}\!\sim$$

We show that Invariant 1(c) holds for P, by noting that $(\frac{1}{c} \cdot (c \cdot p_1 - p_2 + p_2))\!\downarrow = p_1$. Thus, we have $T(P) : \varphi_1^p$ as required by property (i).

$\perp$ **Case** Consider when P is a proof of the following form:

$$\frac{P_1 : \varphi_1 \Leftrightarrow \varphi_2 \quad P_2 : \varphi_2 \Leftrightarrow \perp}{P : \varphi_1 \Leftrightarrow \perp}\ \textsf{iff\_trans}$$

By assumption of Invariant 1(b) for P_1, we have that $T(P_1) : (c \cdot p_1 - p_2)\!\downarrow = 0$, where $\varphi_1^p ::= (p_1 \sim 0)$ and $\varphi_2^p ::= (p_2 \sim 0)$. Note that property (iv) of Invariant 1(b) guarantees that a multiplication of an equation of the form $p(\nsim)_\downarrow 0$ by the constant $\frac{1}{c}$ is legal. Assume Invariant 1(d) holds for P_2. The following defines the transformed proof $T(P)$:

$$\cfrac{\cfrac{T(P_2) : (-p_2)\!\downarrow\!(\nsim)_\downarrow 0 \quad T(P_1) : (c \cdot p_1 - p_2)\!\downarrow = 0}{(-p_2 - (c \cdot p_1 - p_2))\!\downarrow\!(\nsim)_\downarrow 0}\ \textsf{lra\_sub}(\nsim)_\downarrow =}{(\frac{1}{c} \cdot (-p_2 - (c \cdot p_1 - p_2)))\!\downarrow\!(\nsim)_\downarrow 0}\ \textsf{lra\_mult\_c}(\nsim)_\downarrow$$

We show that Invariant 1(d) holds for P, by noting that $(\frac{1}{c} \cdot (-p_2 - (c \cdot p_1 - p_2)))\!\downarrow = (-p_1)\!\downarrow$. Thus, we have $T(P) : (\neg\varphi_1)^p$ as required by property (i).

Default Case Consider when P is a proof of the following form:

$$\frac{P_1 : \varphi_1 \Leftrightarrow \varphi_2 \quad P_2 : \varphi_2 \Leftrightarrow \varphi_3}{P : \varphi_1 \Leftrightarrow \varphi_3}\ \textsf{iff\_trans}$$

Assume Invariant 1(b) holds for P_1 and P_2. The following defines the transformed proof $T(P)$:

$$\cfrac{\cfrac{T(P_1) : (c_1 \cdot p_1 - p_2)\!\downarrow = 0}{(c_2 \cdot (c_1 \cdot p_1 - p_2))\!\downarrow = 0}\ \textsf{lra\_mult\_c=} \qquad T(P_2) : (c_2 \cdot p_2 - p_3)\!\downarrow = 0}{T(P) : (c_2 \cdot (c_1 \cdot p_1 - p_2) + (c_2 \cdot p_2 - p_3))\!\downarrow = 0}\ \textsf{lra\_add==}$$

We show that Invariant 1(b) holds for P. Firt note that $(c_2 \cdot (c_1 \cdot p_1 - p_2) + (c_2 \cdot p_2 - p_3))\!\downarrow = ((c_2 \cdot c_1) \cdot p_1 - p_3)\!\downarrow$, giving us property (i). Properties (ii) and (iii) hold by assumption from Invariant 1(b) for P_1 and P_2. To show property (iv), note that $c_1 \neq 0$ and $c_2 \neq 0$ imply $c_2 \cdot c_1 \neq 0$, and similarly for $>$.

6.2.3 iff_mp

⊥ Case Consider when P is a proof of the following form:

$$\frac{P_1 : \varphi_1 \quad P_2 : \varphi_1 \Leftrightarrow \bot}{P : \bot}\ \textsf{iff\_mp}$$

Assume Invariant 1(a) holds for P_1 and Invariant 1(d) holds for P_2. The following defines the transformed proof $T(P)$:

$$\cfrac{\cfrac{T(P_1) : p \sim 0 \quad T(P_2) : (-p){\downarrow}(\nsim)_{\downarrow}0}{(p + -p){\downarrow} \sim' 0}\ \textsf{lra\_add}{\sim}\ (\nsim)_{\downarrow}}{T(P) : \bot}\ \textsf{lra\_contra}{\sim'}$$

where $\sim'$ is $(\sim \cdot(\nsim)_{\downarrow})$.

Note that since $\sim'$ is $(\sim \cdot(\nsim)_{\downarrow})$, $\sim'$ is restricted to be one of $>, \neq$. Thus, in both cases we have a contradiction from $0 \sim' 0$, and Invariant 1(e) holds for P.

Default Case Consider when P is a proof of the following form:

$$\frac{P_1 : \varphi_1 \quad P_2 : \varphi_1 \Leftrightarrow \varphi_2}{P : \varphi_2}\ \textsf{iff\_mp}$$

Assume Invariant 1(a) holds for P_1. By assumption of Invariant 1(b) for P_2, we have that $\varphi_2^p ::= p_2 \sim 0$. The following defines the transformed proof $T(P)$:

$$\cfrac{\cfrac{T(P_1) : p_1 \sim 0}{(c \cdot p_1){\downarrow} \sim 0}\ \textsf{lra\_mult\_c}{\sim} \quad T(P_2) : (c \cdot p_1 - p_2){\downarrow} = 0}{T(P) : (c \cdot p_1 - (c \cdot p_1 - p_2)){\downarrow} \sim 0}\ \textsf{lra\_sub}{\sim}{=}$$

We show that Invariant 1(a) holds for P, noting that $(c{\cdot}p_1 - (c{\cdot}p_1 - p_2)){\downarrow} = p_2$. Thus, $T(P) : \varphi_2^p$, as required by property (i).

6.2.4 iff_symm

Consider when P is a proof of the following form:

$$\frac{P_1 : \varphi_1 \Leftrightarrow \varphi_2}{P : \varphi_2 \Leftrightarrow \varphi_1}\ \textsf{iff\_symm}$$

By assumption of Invariant 1(b) for P_1, we have that $T(P_1) : (c \cdot p_1 - p_2){\downarrow} = 0$, where $\varphi_1^p ::= (p_1 \sim 0)$ and $\varphi_2^p ::= (p_2 \sim 0)$. The following defines the transformed proof $T(P)$:

$$\frac{T(P_1) : (c \cdot p_1 - p_2){\downarrow} = 0}{T(P) : (-\frac{1}{c} \cdot (c \cdot p_1 - p_2)){\downarrow} = 0} \ \textsf{lra\_mult\_c}_{=}$$

We show that Invariant 1(b) holds for P. Note that $(-\frac{1}{c} \cdot (c \cdot p_1 - p_2)){\downarrow} = (\frac{1}{c} \cdot p_2 - p_1){\downarrow}$, giving us property (i). Properties (ii) and (iii) hold by assumption of Invariant 1(b) for P_1. To show property (iv), note that $c \sim 0$ implies $\frac{1}{c} \sim 0$ when $\sim$ is $>$ or $\neq$.

6.2.5 basic_subst_op

Consider when P is a proof of the following form:

$$\frac{P_1 : t_1 = t_2 \quad P_2 : t_3 = t_4}{P : t_1 \sim t_3 \Leftrightarrow t_2 \sim t_4} \ \textsf{basic\_subst\_op}$$

Assume Invariant 1(a) holds for P_1 and P_2. The following defines the transformed proof $T(P)$, when $\sim \in \{=, >, \geq\}$:

$$\frac{T(P_1) : (t_1 - t_2){\downarrow} = 0 \quad T(P_2) : (t_3 - t_4){\downarrow} = 0}{T(P) : ((t_1 - t_2) - (t_3 - t_4)){\downarrow} = 0} \ \textsf{lra\_sub==}$$

We show that Invariant 1(b) holds for P. Note that $((t_1 - t_2) - (t_3 - t_4)){\downarrow} = (1 \cdot (t_1 - t_3){\downarrow} - (t_2 - t_4){\downarrow}){\downarrow}$, giving us property (i). By unfolding definitions, we have properties (ii) and (iii). Property (iv) is satisfied by the constant 1 for both cases of $>, \neq$.

The following defines the transformed proof $T(P)$, when $\sim \in \{\neq, <, \leq\}$:

$$\frac{T(P_2) : (t_3 - t_4){\downarrow} = 0 \quad T(P_1) : (t_1 - t_2){\downarrow} = 0}{T(P) : ((t_3 - t_4) - (t_1 - t_2)){\downarrow} = 0} \ \textsf{lra\_sub==}$$

We show that Invariant 1(b) holds for P. Note that $((t_3 - t_4) - (t_1 - t_2)){\downarrow} = (1 \cdot (t_3 - t_1){\downarrow} - (t_4 - t_2){\downarrow}){\downarrow}$, giving us property (i). By unfolding definitions, we have properties (ii) and (iii). Property (iv) is satisfied by the constant 1 for both cases of $>, \neq$.

6.2.6 basic_subst_op_1

Addition Case Consider when P is a proof of the following form:

$$\frac{P_1 : t_1 = t_2 \quad P_2 : t_3 = t_4}{P : t_1 + t_3 = t_2 + t_4} \ \textsf{basic\_subst\_op\_1}$$

Assume Invariant 1(a) holds for P_1 and P_2. The following defines the transformed proof $T(P)$:

$$\frac{T(P_1) : (t_1 - t_2)\downarrow = 0 \quad T(P_2) : (t_3 - t_4)\downarrow = 0}{T(P) : ((t_1 - t_2) + (t_3 - t_4))\downarrow = 0} \ \textsf{lra\_add==}$$

We show that Invariant 1(a) holds for P, by noting that $((t_1 - t_2) + (t_3 - t_4))\downarrow = ((t_1 + t_3) - (t_2 + t_4))\downarrow$. Thus, $T(P) : (t_1 + t_3 = t_2 + t_4)^p$, as required by property (i).

Subtraction Case The case for proving the subtraction case is very similar to the addition case, where instead of **lra_add==** , we use the LFSC rule **lra_sub==** .

Multiplication Case Consider when P is a proof of the following form:

$$\frac{P_1 : c = c \quad P_2 : t_1 = t_2}{P : c \cdot t_1 = c \cdot t_2} \ \textsf{basic\_subst\_op\_1}$$

Assume Invariant 1(a) holds for P_2. The following defines the transformed proof $T(P)$:

$$\frac{T(P_2) : (t_1 - t_2)\downarrow = 0}{T(P) : (c \cdot (t_1 - t_2))\downarrow = 0} \ \textsf{lra\_mult\_c}_{=}$$

We show that Invariant 1(a) holds for P, noting that $(c \cdot (t_1 - t_2))\downarrow = ((c \cdot t_1) - (c \cdot t_2))\downarrow$. Thus, $T(P) : (c \cdot t_1 = c \cdot t_2)^p$, as required by property (i).

6.2.7 basic_subst_op_0

Not Case Consider when P is a proof of the following forms:

$$\frac{P_1 : t_1 \sim t_2 \Leftrightarrow t_3 \sim t_4}{P : \neg(t_1 \sim t_2) \Leftrightarrow \neg(t_3 \sim t_4)} \ \textsf{basic\_subst\_op\_0}$$

Assume that Invariant 1(b) holds for P_1. The following defines the transformed proof $T(P)$:

$$\frac{T(P_1) : (c \cdot p_1 - p_2)\downarrow = 0}{T(P) : (-1 \cdot (c \cdot p_1 - p_2))\downarrow = 0} \ \textsf{lra\_mult\_c}_{=}$$

We show that Invariant 1(b) holds for P. Note that $(-1 \cdot (c \cdot p_1 - p_2))\downarrow = (c \cdot (-p_1) - (-p_2))\downarrow$, giving us property (i). By unfolding our definitions and relying upon Lemma 2 (twice), we have properties (ii) and (iii). Property (iv) holds as a consequence of our assumption of Invariant 1(b) property (iv) for P_1.

Unary Minus Case Consider when P is a proof of the following forms:

$$\frac{P_1 : t_1 = t_2}{P : -t_1 = -t_2}\ \textsf{basic\_subst\_op\_0}$$

Assume that Invariant 1(a) holds for P_1. The following defines the transformed proof $T(P)$:

$$\frac{T(P_1) : (t_1 - t_2)\!\downarrow = 0}{T(P) : (-1 \cdot (t_1 - t_2))\!\downarrow = 0}\ \textsf{lra\_mult\_c}_{=}$$

We show that Invariant 1(a) holds for P, noting that $(-1 \cdot (t_1 - t_2))\!\downarrow = ((-t_1) - (-t_2))\!\downarrow$. Thus, $T(P) : (-t_1 = -t_2)^p$, as required by property (i).

6.2.8 eq_trans

Consider when P is a proof of the following form:

$$\frac{P_1 : t_1 = t_2 \quad P_2 : t_2 = t_3}{P : t_1 = t_3}\ \textsf{eq\_trans}$$

Assume Invariant 1(a) holds for P_1 and P_2. The following defines the transformed proof $T(P)$:

$$\frac{T(P_1) : (t_1 - t_2)\!\downarrow = 0 \quad T(P_2) : (t_2 - t_3)\!\downarrow = 0}{T(P) : ((t_1 - t_2) + (t_2 - t_3))\!\downarrow = 0}\ \textsf{lra\_add}{=}{=}$$

We show that Invariant 1(a) holds for P, noting that $((t_1 - t_2) + (t_2 - t_3))\!\downarrow = (t_1 - t_3)\!\downarrow$. Thus, $T(P) : (t_1 = t_3)^p$, as required by property (i).

6.2.9 eq_symm

Consider when P is a proof of the following form:

$$\frac{P_1 : t_1 = t_2}{P : t_2 = t_1}\ \textsf{eq\_symm}$$

Assume Invariant 1(a) holds for P_1. The following defines the transformed proof $T(P)$:

$$\frac{T(P_1) : (t_1 - t_2)\!\downarrow = 0}{T(P) : (-1 \cdot (t_1 - t_2))\!\downarrow = 0}\ \textsf{lra\_mult\_c}_{=}$$

We show that Invariant 1(a) holds for P, noting that $(-1 \cdot (t_1 - t_2))\!\downarrow = (t_2 - t_1)\!\downarrow$. Thus, $T(P) : (t_2 = t_1)^p$, as required by property (i).

6.2.10 real_shadow

Consider when P is a proof of the following form:

$$\frac{P_1 : t_1 \prec_1 t_2 \quad P_2 : t_2 \prec_2 t_3}{P : t_1 \prec' t_3} \ \mathsf{real\_shadow}$$

where $\prec'$ is $(\prec_1 \cdot \prec_2)$.

Assume Invariant 1(a) holds for P_1 and P_2. The following defines the transformed proof $T(P)$:

$$\frac{T(P_1) : (t_2 - t_1)\!\downarrow\, \succ_1 0 \quad T(P_2) : (t_3 - t_2)\!\downarrow\, \succ_2 0}{T(P) : ((t_2 - t_1) + (t_3 - t_2))\!\downarrow\, \succ' 0} \ \mathsf{lra\_add}{\succ_1}{\succ_2}$$

We show that Invariant 1(a) holds for P, noting that $((t_2 - t_1) + (t_3 - t_2))\!\downarrow = (t_3 - t_1)\!\downarrow$. Thus, $T(P) : (t_1 \prec' t_3)^p$, as required by property (i).

6.2.11 real_shadow_eq

Consider when P is a proof of the following form:

$$\frac{P_1 : t_1 \leq t_2 \quad P_2 : t_2 \leq t_1}{P : t_1 = t_2} \ \mathsf{real\_shadow\_eq}$$

Assume Invariant 1(a) holds for P_1 and P_2. The following defines the transformed proof $T(P)$:

$$\frac{T(P_1) : (t_2 - t_1)\!\downarrow\, \geq 0 \quad T(P_2) : (t_1 - t_2)\!\downarrow\, \geq 0}{T(P) : (t_1 - t_2)\!\downarrow\, = 0} \ \mathsf{lra\_\geq\_\geq\_to\_=}$$

We have that Invariant 1(a) holds for P, noting that $T(P) : (t_1 = t_2)^p$, as required by property (i).

6.2.12 add_inequalities

Consider when P is a proof of the following form:

$$\frac{P_1 : t_1 \prec_1 t_2 \quad P_2 : t_3 \prec_2 t_4}{P : t_1 + t_3 \prec' t_2 + t_4} \ \mathsf{add\_inequalities}$$

where $\prec'$ is $(\prec_1 \cdot \prec_2)$.

Assume Invariant 1(a) holds for P_1 and P_2. The following defines the transformed proof $T(P)$:

$$\frac{T(P_1) : (t_2 - t_1)\!\downarrow\ \succ_1 0 \quad T(P_2) : (t_4 - t_3)\!\downarrow\ \succ_2 0}{T(P) : ((t_2 - t_1) + (t_4 - t_3))\!\downarrow\ \succ' 0}\ \textsf{lra\_add}{\succ_1}{\succ_2}$$

We show that Invariant 1(a) holds for P, noting that $((t_2 - t_1) + (t_4 - t_3))\!\downarrow\ = ((t_2 + t_4) - (t_1 + t_3))\!\downarrow$. Thus, $T(P) : (t_1 + t_3 \prec' t_2 + t_4)^p$, as required by property (i).

6.2.13 optimized_subst_op

+ Case Consider when P is a proof of the following form:

$$\frac{P_1 : t_1 = s_1 \quad \ldots \quad P_n : t_n = s_n}{P : t_1 + \ldots + t_n = s_1 + \ldots + s_n}\ \textsf{optimized\_subst\_op}$$

Assume Invariant 1(a) holds for $P_1 \ldots P_n$. The following defines the transformed proof $T(P)$:

$$\cfrac{\cfrac{T(P_1) : p_1 = 0 \quad T(P_2) : p_2 = 0}{(p_1 + p_2)\!\downarrow\ = 0}\ \textsf{lra\_add==} \qquad \vdots \qquad T(P_n) : p_n = 0}{T(P) : (p_1 + \ldots + p_n)\!\downarrow\ = 0}\ \textsf{lra\_add==}$$

where $p_i = (t_i - s_i)\!\downarrow$ for all i.

We show that Invariant 1(a) holds for P, by noting that $(p_1 + \ldots + p_n)\!\downarrow\ = ((t_1 - s_1) + \ldots + (t_n - s_n))\!\downarrow\ = (t_1 + \ldots + t_n - (s_1 + \ldots + s_n)\!\downarrow$. Thus, $T(P) : (t_1 + \ldots + t_n = s_1 + \ldots + s_n)^p$, as required by property (i).

Other Cases A similar translation can be used for when P is a proof of the following form:

$$\frac{P_1 : t_1 = s_1 \quad \ldots \quad P_n : t_n = s_n}{P : t_1 \bowtie \ldots \bowtie t_n = s_1 \bowtie \ldots \bowtie s_n}\ \textsf{optimized\_subst\_op}$$

where $\bowtie \in \{-, \cdot\}$.

6.2.14 cycleConflict

Consider when P is a proof of the following form:

$$\frac{P_1 : t_1 \prec_1 t_1' \quad \ldots \quad P_n : t_n \prec_n t_n'}{\bot}\ \textsf{cycleConflict}$$

where $(t_1 \prec_1 t'_1) \wedge \ldots \wedge (t_n \prec_n t'_n) \Rightarrow \bot$.

Assume Invariant 1(a) holds for $P_1 \ldots P_n$. The following defines the transformed proof $T(P)$:

$$\dfrac{\dfrac{\dfrac{T(P_1) : p_1 \succ_1 0 \quad T(P_2) : p_2 \succ_2 0}{(p_1 + p_2){\downarrow} \succ'_2 0}\ \mathsf{lra\_add}{\succ_1}{\succ_2} \quad \vdots \qquad T(P_n) : p_n \succ_n 0}{(p_1 + \ldots + p_n){\downarrow} \succ'_{n-1} 0}\ \mathsf{lra\_add}{\succ'_{n-1}}{\succ_n}}{T(P) : \bot}\ \mathsf{lra\_contra}{\succ'_n}$$

where $P'_i = T_p(T_{\mathsf{LRA1}}(P_i))$, $\succ'_1$ is $\succ_1$, and $\succ'_i$ is $(\succ'_{i-1} \cdot \succ_i)$ for $i > 1$.

Due to the restricted form for the premises used in the **cycleConflict** rule (as discussed in section 5.3), the summation $(p_1 + \ldots + p_n){\downarrow} \succ'_{n-1} 0 ::= 0 > 0$, giving us a contradiction. Thus, we have that Invariant 1(e) holds for P.

6.2.15 const_pred

$\bot$ **Case** Consider when P is a proof of the following form:

$$\dfrac{\{0 \nsim c\}}{P : (0 \sim c) \Leftrightarrow \bot}\ \mathsf{const\_pred}_1$$

The following defines the transformed proof $T(P)$:

$$\dfrac{}{T(P) : c(\nsim)_{\downarrow} 0}\ \mathsf{lra\_axiom\_}(\nsim)_{\downarrow}$$

when $\sim \in \{=, >, \geq\}$

$$\dfrac{}{T(P) : (-c){\downarrow} \nsim 0}\ \mathsf{lra\_axiom\_}\nsim$$

when $\sim \in \{\neq, <, \leq\}$

In both cases, we have that Invariant 1(d) holds for P, noting that $T(P) : (\neg(0 \sim c))^p$ as required by property (i).

$\top$ **Case** Consider when P is a proof of the following form:

$$\dfrac{\{0 \sim c\}}{P : (0 \sim c) \Leftrightarrow \top}\ \mathsf{const\_pred}_2$$

The following defines the transformed proof $T(P)$ when $\sim \in \{=, >, \geq\}$:

$$\frac{}{T(P) : (-c)\!\downarrow \sim 0}\ \mathsf{lra\_axiom\_\sim}$$

when $\sim \in \{=, >, \geq\}$

$$\frac{}{T(P) : c \sim_{\downarrow} 0}\ \mathsf{lra\_axiom\_\sim_{\downarrow}}$$

when $\sim \in \{\neq, <, \leq\}$

In both cases, we have that Invariant 1(c) holds for P, noting that $T(P) : (0 \sim c)^p$ as required by property (i).

6.2.16 Equality Axioms

In this section, we consider when P is a proof of any of the following forms:

$$\frac{}{P : t_1 = t_1}\ \mathsf{refl} \qquad \frac{}{P : -t = (-1) \cdot t}\ \mathsf{uminus\_to\_mult}$$

$$\frac{}{P : (t_1 - t_2) = t_1 + (-1 \cdot t_2)}\ \mathsf{minus\_to\_plus} \qquad \frac{\{t' \text{ canonical form of } t\}}{P : t = t'}\ \mathsf{canon}$$

In all of the above cases, we define $T(P)$ as:

$$\frac{}{T(P) : 0 = 0}\ \mathsf{lra\_axiom}_{=}$$

We show that Invariant 1(a) holds for each case of $P : t = t'$, noting we have that $(t - t')\!\downarrow = 0$ in each case. Thus, $T(P) : (t = t')^p$, giving us property (i) as required.

6.2.17 Rewrite Axioms

In this section, we consider when P is a proof of any of the following forms:

$$\frac{\{c \text{ positive}\}}{P : t_1 \prec t_2 \Leftrightarrow c \cdot t_1 \prec c \cdot t_2}\ \mathsf{mult\_ineqn} \qquad \frac{\{c \text{ non-zero}\}}{P : t_1 = t_2 \Leftrightarrow c \cdot t_1 = c \cdot t_2}\ \mathsf{mult\_eqn}$$

$$\frac{}{P : t_1 \sim t_2 \Leftrightarrow 0 \sim t_2 - t_1}\ \mathsf{right\_minus\_left} \qquad \frac{}{P : t_1 \sim t_2 \Leftrightarrow t_1 + t_3 \sim t_2 + t_3}\ \mathsf{plus\_pred}$$

$$\frac{}{P : t_1 \succ t_2 \Leftrightarrow t_2 \prec t_1}\ \mathsf{flip\_ineq} \qquad \frac{}{P : \neg(t_1 \prec t_2) \Leftrightarrow t_1 \not\prec t_2}\ \mathsf{negated\_ineq}$$

$$\frac{}{P : (\neg\neg(t_1 \sim t_2)) \Leftrightarrow (t_1 \sim t_2)}\ \mathsf{rewrite\_not\_not} \qquad \frac{}{P : t_1 = t_2 \Leftrightarrow t_2 = t_1}\ \mathsf{rewrite\_eq\_symm}$$

$$\frac{}{P : (t = t) \Leftrightarrow \top}\ \mathsf{rewrite\_eq\_refl}$$

In all of the above cases, we define $T(P)$ as:

$$\frac{}{T(P) : 0 = 0}\ \mathsf{lra\_axiom}_{=}$$

We show that Invariant 1 holds for each case of $P : \varphi_1 \Leftrightarrow \varphi_2$.

Invariant 1(b) holds for mult_ineqn. Note we have that $\varphi_1^p ::= p \succ 0$ where $p = (t_2 - t_1)\downarrow$, giving us property (ii). We have property (iii), further noting that $\varphi_2^p ::= (c \cdot p)\downarrow \succ 0$. Thus, $T(P) : (c \cdot p - (c \cdot p))\downarrow = 0$, giving us property (i). Property (iv) is satisfied as a result of the Cvc3 condition that c is positive.

Similarly for mult_eqn, we have that $\varphi_1^p ::= p = 0$ where $p = (t_1 - t_2)\downarrow$, giving us property (ii), $\varphi_2^p ::= (c \cdot p)\downarrow = 0$ for property (iii), and $T(P) : (c \cdot p - (c \cdot p))\downarrow = 0$ for property (i). Property (iv) is satisfied as a result of the Cvc3 condition that c is non-zero.

Invariant 1(c) holds for rewrite_eq_refl. Note that $(t - t)\downarrow = 0$, thus, $T(P) : (t = t)^p$ as required by property (i).

Invariant 1(b) holds for all the other cases of $P : \varphi_1 \Leftrightarrow \varphi_2$. Note we have that $\varphi_1^p ::= \varphi_2^p ::= p \sim 0$ (that is, they are identical) for some polynomial p, giving us properties (ii) and (iii). Thus, we have that $T(P) : (1 \cdot p - p)\downarrow = 0$, giving us property (i). Property (iv) is satisfied by the constant 1 for both cases of $>, \neq$.

6.2.18 Miscellaneous Propositional Rules

In this section, we consider when P is a proof of any of the following forms:

$$\frac{P_1 : (t_1 \sim t_2)}{P : (t_1 \sim t_2) \Leftrightarrow \top}\ \textsf{iff\_true} \qquad \frac{P_1 : (t_1 \sim t_2) \Leftrightarrow \top}{P : (t_1 \sim t_2)}\ \textsf{iff\_true\_elim}$$

$$\frac{P_1 : (t_1 \sim t_2)}{P : \neg(t_1 \sim t_2) \Leftrightarrow \bot}\ \textsf{iff\_not\_false} \qquad \frac{P_1 : (t_1 \sim t_2) \Leftrightarrow \bot}{P : \neg(t_1 \sim t_2)}\ \textsf{iff\_false\_elim}$$

$$\frac{P_1 : \neg(t_1 \sim t_2)}{P : (t_1 \sim t_2) \Leftrightarrow \bot}\ \textsf{not\_to\_iff} \qquad \frac{P_1 : \neg\neg(t_1 \sim t_2)}{P : (t_1 \sim t_2)}\ \textsf{not\_not\_elim}$$

Assume that Invariant 1 holds for P_1. In all of the above cases, we define $T(P) = T(P_1)$. It can be shown that Invariant 1 holds for all cases of P, each coming as a direct consequence of Invariant 1 holding for P_1.

6.3 Patch Operator T_p^{-1}

Define a proof translation function $T_p^{-1} : \mathbf{P} \rightarrow \mathbf{P}$ from $\mathcal{L}$ proofs to $\mathcal{L}$ proofs with the following property:

Lemma 6

For all $\mathcal{C}$ theory reasoning proofs $P : \Gamma \vdash \varphi$, we have $T_p^{-1}(T(P)) : \Gamma \vdash \varphi$.

In the following section, we will give the definition $T_p^{-1}(T(P))$ for an arbitrary theory reasoning $\mathcal{C}$ proof P. [6] We will use three additional proof rules from the $\mathcal{L}$ calculus in our definition:

[6]For all other $\mathcal{L}$ proofs P', we define $T_p^{-1}(P') = P'$.

$$\dfrac{\begin{matrix}[\neg\varphi]\\ \vdots\\ \bot\end{matrix}}{\varphi}\ \textsf{proof\_by\_contradiction} \qquad \dfrac{\begin{matrix}[\varphi_1]\\ \vdots\\ \varphi_2\end{matrix}}{\varphi_1 \Rightarrow \varphi_2}\ \textsf{impl\_intro}$$

$$\dfrac{\varphi_1 \Rightarrow \varphi_2 \quad \varphi_2 \Rightarrow \varphi_1}{\varphi_1 \Leftrightarrow \varphi_2}\ \textsf{iff\_intro}$$

By Lemma 5, we know that Invariant 1 holds for P. In order to define $T_p^{-1}(T(P))$, we will case split on Invariant 1 for P and show that Lemma 6 holds in each case.

6.3.1 Case (a)

Assume that $P : \Gamma \vdash \varphi$ and Invariant 1(a) holds for P. By property (i) of Invariant 1(a), $T(P) : \Gamma \vdash \varphi^p$. We define $T_p^{-1}(T(P))$ as the following: [7]

$$\dfrac{\dfrac{\dfrac{T(P) : \Gamma_1 \vdash \varphi^p \quad T_p(P') : \Gamma_1 \vdash (\neg\varphi)^p}{\Gamma_1 \vdash (p + -p)\!\downarrow\, \sim' 0}\ \textsf{lra\_add}\!\sim (\nsim)_\downarrow}{\Gamma_1 \vdash \bot}\ \textsf{lra\_contra}\!\sim'}{T_p^{-1}(T(P)) : \Gamma \vdash \varphi}\ \textsf{proof\_by\_contradiction}$$

where $\varphi^p ::= p \sim 0$, P' is an assertion of $\neg\varphi$, and $\Gamma_1 = \Gamma \cup \neg\varphi$.

We use proof by contradiction, where we conclude φ by proving a contradiction under the assumption $\neg\varphi$ asserted by P'. We convert this assumption to a polynomial formula with the proof translation T_p to obtain proof $T_p(P')$. Lemma 2 tell us that $(\neg\varphi)^p ::= (-p)\!\downarrow\!(\nsim)_\downarrow 0$. We then add this formula with $p \sim 0$, as proven by $T(P)$, to produce the inconsistent formula $(p + -p)\!\downarrow\, \sim' 0$. Note that $\sim'$ is $(\sim \cdot (\nsim)_\downarrow)$, which restricts $\sim'$ to be one of $\neq, >$. Thus $(p + -p)\!\downarrow\, = 0 \sim' 0$ gives a contradiction for both cases of $\sim'$.

6.3.2 Case (b)

Assume that $P : \Gamma \vdash \varphi_1 \Leftrightarrow \varphi_2$ and Invariant 1(b) holds for P. Thus, we have that $T(P) : (c \cdot p_1 - p_2)\!\downarrow\, = 0$, $\varphi_1^p ::= p_1 \sim 0$ and $\varphi_2^p ::= p_2 \sim 0$ for some polynomials p_1 and p_2.

We must prove both directions of the double implication $\varphi_1 \Leftrightarrow \varphi_2$. We give the proof of $\Rightarrow$ (call it P_1): [8]

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{T_p(P_1') : \Gamma_2 \vdash \varphi_1^p}{\Gamma_2 \vdash (c \cdot p_1)\!\downarrow\, \sim 0}\ 1 \quad T_p(P_2') : \Gamma_2 \vdash (\neg\varphi_2)^p}{\Gamma_2 \vdash (c \cdot p_1 + -p_2)\!\downarrow\, \sim' 0}\ 2 \quad T(P) : \Gamma_2 \vdash p = 0}{\Gamma_2 \vdash (p - p)\!\downarrow\, \sim' 0}\ 3}{\Gamma_2 \vdash \bot}\ \textsf{lra\_contra}\!\sim'}{\Gamma_1 \vdash \varphi_2}\ \textsf{proof\_by\_contradiction}}{P_1 : \Gamma \vdash \varphi_1 \Rightarrow \varphi_2}\ \textsf{impl\_intro}$$

[7] We (implicitly) weaken $T(P) : \Gamma \vdash \varphi^p$ to $T(P) : \Gamma_1 \vdash \varphi^p$ for the sake of consistency.

[8] We (implicitly) weaken $T(P) : \Gamma \vdash p = 0$ to $T(P) : \Gamma_2 \vdash p = 0$ for the sake of consistency.

where **1** is lra_multc$\sim$, **2** is lra_add$\sim$ $(\nsim)_{\downarrow}$, **3** is lra_sub$\sim'$=, P_1' is an assertion of φ_1, P_2' is an assertion of $\neg\varphi_2$, $p = (c \cdot p_1 - p_2)\!\downarrow$, $\Gamma_1 = \Gamma \cup \varphi_1$, and $\Gamma_2 = \Gamma_1 \cup \neg\varphi_2$.

We use the implication introduction rule to introduce the assumption φ_1, and then use proof by contradiction to introduce the assumption $\neg\varphi_2$. These are converted to polynomial formulas with the proof translation T_p to obtain proofs $T_p(P_1')$ and $T_p(P_2')$. We multiply p_1 by c to obtain $(c \cdot p_1)\!\downarrow \sim 0$, noting that property (iv) of Invariant 1(b) guarantees that is a legal constant. By Lemma 2, we know $(\neg\varphi_2)^p ::= (-p_2)\!\downarrow\!(\nsim)_{\downarrow}0$. Thus, we add these formulas to obtain the equation $(c \cdot p_1 + -p_2)\!\downarrow \sim' 0$, where $\sim'$ is $(\sim \cdot (\nsim)_{\downarrow})$. Similar to Case (a), this restricts $\sim'$ to be one of $\neq, >$. Finally, we subtract $p = 0$, as proven by $T(P)$, to obtain $(p - p)\!\downarrow = 0 \sim' 0$, giving a contradiction for both cases of $\sim'$.

A similar proof, call it P_2, gives us the $\Leftarrow$ direction. Because these two proofs both involve the subproof $T(P)$, we introduce $T(P)$ as a lemma and reference it in both instances. [9] Overall, we define $T_p^{-1}(T(P))$ to be:

$$\dfrac{P_1 : \Gamma \vdash \varphi_1 \Rightarrow \varphi_2 \quad P_2 : \Gamma \vdash \varphi_2 \Rightarrow \varphi_1}{T_p^{-1}(T(P)) : \Gamma \vdash \varphi_1 \Leftrightarrow \varphi_2} \;\textsf{iff\_intro}$$

6.3.3 Case (c)

When $P : \varphi \Leftrightarrow \top$, and Invariant 1(c) holds for P, the proof is similar to Case (a) in the $\Leftarrow$ direction of the implication, and trivial in the $\Rightarrow$ direction.

6.3.4 Case (d)

When $P : \varphi \Leftrightarrow \bot$, and Invariant 1(d) holds for P, the proof is similar to Case (a) in the $\Rightarrow$ direction of the implication, and trivial in the $\Leftarrow$ direction.

6.3.5 Case (e)

When $P : \bot$, and Invariant 1(e) holds for P (that is, $T(P)$ also concludes $\bot$), we define $T_p^{-1}(T(P)) = T(P)$. $\square$

We are now ready to define the aggressive liberal translation $T_{\textsf{LRA2a}}$.

Translation $T_{\textsf{LRA2a}}$ Define a proof translation operator $T_{\textsf{LRA2a}} : \mathbf{P} \rightarrow \mathbf{P}$ from $\mathcal{C}$ proofs to $\mathcal{L}$ proofs. Say P is a proof in the following form:

$$\dfrac{P_1 : \varphi_1 \quad \ldots \quad P_n : \varphi_n}{P : \varphi} \;\textsf{r}$$

If all premises $P_1 \ldots P_n$ can be compacted according to T, and further T is defined for rule r, we continue applying polynomial compaction to P. Otherwise, we will revert all proofs of our premises and subsequently use the literal translation LRA1.

[9] In many cases, it suffices for us to conclude $\varphi_1 \Rightarrow \varphi_2$ only. For optimization purposes, such cases are recognized by our translation.

Formally, define T_{LRA2a} as the following:

For all theory reasoning $\mathcal{C}$ proofs P,

$T_{\mathsf{LRA2a}}(P) = T(P)$.

Otherwise,

$T_{\mathsf{LRA2a}}(P) = T_{\mathsf{LRA1}}((\{T_p^{-1}(T_{\mathsf{LRA2a}}(P_1)), \ldots, T_p^{-1}(T_{\mathsf{LRA2a}}(P_n)\}, r, \varphi))$.

7 Liberal Conversion

Translation T_{LRA2} The translation function $T_{\mathsf{LRA2}} : \mathbf{P} \rightarrow \mathbf{P}$ can be described in terms of the translations T_{LRA1} and T_{LRA2a}.

7.1 Learned Clause

Say P is the following proof of the following form:

$$\frac{P_1 : \varphi_1, \ldots, \varphi_n \vdash \bot}{P : \vdash \neg\varphi_1 \vee \ldots \vee \neg\varphi_n} \ \textsf{learned\_clause}$$

In this case, $T_{\mathsf{LRA2}}(P) = T_{\mathsf{LRA2a}}(P)$.

7.2 Default Case

For all other proofs $P = (\{P_1, \ldots, P_n\}, r, \varphi)$,
$T_{\mathsf{LRA2}}(P) = T_{\mathsf{LRA1}}((\{T_{\mathsf{LRA2}}(P_1), \ldots, T_{\mathsf{LRA2}}(P_n)\}, r, \varphi))$. □

We claim that all subproofs P of theory lemmas (whose top node is an instance of the learned_clause rule) are such that $T(P)$ is defined. Because of this, we know that T_p^{-1} is never called as a sub-routine of T_{LRA2}. That is to say, we do not incur any overhead as a result of converting from polynomial formulas to term formulas.

8 Compression for Polynomial Proofs

In the following section, we will discuss all post-processing performed on polynomial portions of the LFSC proof. It is important to note that such processing occurs *at the same time* the proof is created, that is to say, all Cvc3 to LFSC translations incorporate this compression and are done in one pass.

8.1 Trivial Addition

Say we generate the following $\mathcal{L}$ proof P:

$$\frac{P_1 : p = 0 \quad P_2 : 0 = 0}{P : (p + 0)\downarrow = 0} \ \textsf{lra\_add==}$$

In this case, we will return the proof P_1.

Similarly, if we generate the $\mathcal{L}$ proof P:

$$\frac{P_1 : 0 = 0 \quad P_2 : p \sim 0}{P : (0 + p)\!\downarrow\, \sim 0}\ \textsf{lra\_add=}\!\sim$$

In this case, we will return the proof P_2.

8.2 Trivial Subtraction

Say we generate the following $\mathcal{L}$ proof P:

$$\frac{P_1 : p \sim 0 \quad P_2 : 0 = 0}{P : (p - 0)\!\downarrow\, \sim 0}\ \textsf{lra\_sub}\!\sim\!=$$

In this case, we will return the proof P_1.

If we generate the $\mathcal{L}$ proof P:

$$\frac{P_1 : 0 = 0 \quad P_2 : p = 0}{P : (0 - p)\!\downarrow\, = 0}\ \textsf{lra\_sub==}$$

We will return the following proof:

$$\frac{P_2 : p = 0}{P : (-1 \cdot p)\!\downarrow\, = 0}\ \textsf{lra\_multc}\!\sim$$

8.3 Trivial Multiplication

Say we generate the following $\mathcal{L}$ proof P:

$$\frac{P_1 : p \sim 0}{P : (1 \cdot p)\!\downarrow\, \sim 0}\ \textsf{lra\_multc}\!\sim$$

In this case, we will return the proof P_1.

8.4 Repeated Multiplication

Say we generate the following $\mathcal{L}$ proof P:

$$\frac{\dfrac{P_1 : p \sim 0}{(c_1 \cdot p)\!\downarrow\, \sim 0}\ \textsf{lra\_multc}\!\sim}{P : (c_2 \cdot (c_1 \cdot p))\!\downarrow\, \sim 0}\ \textsf{lra\_multc}\!\sim$$

We will return the following proof:

$$\frac{P_1 : p \sim 0}{P : ((c_2 \cdot c_1) \cdot p)\!\downarrow\, \sim 0}\ \textsf{lra\_multc}\!\sim$$

8.5 Addition to Subtraction

Say we generate the following $\mathcal{L}$ proof P:

$$\dfrac{\dfrac{P_1 : p_1 = 0}{(-1 \cdot p_1){\downarrow} = 0}\ \textsf{lra\_multc=} \qquad P_2 : p_2 \sim 0}{P : ((-1 \cdot p_1) + p_2){\downarrow} \sim 0}\ \textsf{lra\_add=}{\sim}$$

We will return the following proof:

$$\dfrac{P_2 : p_2 \sim 0 \quad P_1 : p_1 = 0}{P : (p_2 - p_1){\downarrow} \sim 0}\ \textsf{lra\_sub}{\sim}\textsf{=}$$

A $\mathcal{C}$ Proof Rules

The following is a representative list of rules in the $\mathcal{C}$ calculus. The letters c and t, possibly with subscripts, denote rational constants and arithmetic terms, respectively.

A.1 Core Rules

$$\frac{v \vee \varphi_1 \quad \neg v \vee \varphi_2}{\varphi_1 \vee \varphi_2}\ \textsf{bool\_res} \qquad \frac{\begin{array}{c}[\varphi_1 \wedge \ldots \wedge \varphi_n] \\ \vdots \\ \bot\end{array}}{\neg\varphi_1 \vee \ldots \neg \vee \varphi_n}\ \textsf{learned\_clause}$$

A.2 Rewrite Axioms

$$\frac{\{0 \nsim c\}}{(0 \sim c) \Leftrightarrow \bot}\ \textsf{const\_pred}_1 \qquad \frac{\{0 \sim c\}}{(0 \sim c) \Leftrightarrow \top}\ \textsf{const\_pred}_2$$

$$\frac{\{c \text{ positive}\}}{t_1 \prec t_2 \Leftrightarrow c \cdot t_1 \prec c \cdot t_2}\ \textsf{mult\_ineqn} \qquad \frac{\{c \text{ non-zero}\}}{t_1 = t_2 \Leftrightarrow c \cdot t_1 = c \cdot t_2}\ \textsf{mult\_eqn}$$

$$\frac{}{t_1 \sim t_2 \Leftrightarrow 0 \sim t_2 - t_1}\ \textsf{right\_minus\_left} \qquad \frac{}{t_1 \sim t_2 \Leftrightarrow t_1 + t_3 \sim t_2 + t_3}\ \textsf{plus\_pred}$$

$$\frac{}{t_1 \succ t_2 \Leftrightarrow t_2 \prec t_1}\ \textsf{flip\_ineq} \qquad \frac{}{\neg(t_1 \prec t_2) \Leftrightarrow t_1 \nprec t_2}\ \textsf{negated\_ineq}$$

$$\frac{}{(t = t) \Leftrightarrow \top}\ \textsf{rewrite\_eq\_refl} \qquad \frac{}{t_1 = t_2 \Leftrightarrow t_2 = t_1}\ \textsf{rewrite\_eq\_symm}$$

$$\frac{\{c_1 \prec_1 c_2\}}{0 \prec_2 c_1 + t \Rightarrow 0\ (\prec_1 \cdot \prec_2)\ c_2 + t}\ \textsf{weaker\_ineq}$$

$$\frac{\{c_1 < -c_2\}}{0 \leq c_1 + t_1 \Rightarrow \neg(0 \leq c_2 - t_1)}\ \textsf{imply\_negated\_ineq}$$

A.3 Propositional Rules

$$\frac{\varphi_1 \Leftrightarrow \varphi_2 \quad \varphi_2 \Leftrightarrow \varphi_3}{\varphi_1 \Leftrightarrow \varphi_3}\ \textsf{iff\_trans} \qquad \frac{\varphi_1 \quad \varphi_1 \Leftrightarrow \varphi_2}{\varphi_2}\ \textsf{iff\_mp}$$

$$\frac{\varphi_1 \Leftrightarrow \varphi_2}{\varphi_2 \Leftrightarrow \varphi_1}\ \textsf{iff\_symm} \qquad \frac{}{\varphi \Leftrightarrow \varphi}\ \textsf{iff\_refl}$$

$$\frac{\varphi_1 \Rightarrow \varphi_2 \quad \varphi_2 \Rightarrow \varphi_3}{\varphi_1 \Rightarrow \varphi_3}\ \textsf{impl\_trans} \qquad \frac{\varphi_1 \quad \varphi_1 \Rightarrow \varphi_2}{\varphi_2}\ \textsf{impl\_mp}$$

$$\frac{\varphi}{\varphi \Leftrightarrow \top}\ \textsf{iff\_true} \qquad \frac{\varphi \Leftrightarrow \top}{\varphi}\ \textsf{iff\_true\_elim}$$

$$\frac{\varphi}{\neg\varphi \Leftrightarrow \bot}\ \textsf{iff\_not\_false} \qquad \frac{\varphi \Leftrightarrow \bot}{\neg\varphi}\ \textsf{iff\_false\_elim}$$

$$\frac{\neg\varphi}{\varphi \Leftrightarrow \bot}\ \textsf{not\_to\_iff} \qquad \frac{\neg\neg\varphi}{\varphi}\ \textsf{not\_not\_elim}$$

$$\frac{}{(\neg\neg\varphi) \Leftrightarrow \varphi}\ \textsf{rewrite\_not\_not} \qquad \frac{}{(\varphi_1 \Rightarrow \varphi_2) \Leftrightarrow (\varphi_2 \vee \neg\varphi_1)}\ \textsf{rewrite\_implies}$$

$$\frac{\varphi_1 \Leftrightarrow \varphi_2}{\neg\varphi_1 \Leftrightarrow \neg\varphi_2}\ \textsf{basic\_subst\_op0}_2 \qquad \frac{}{(\varphi_1 \Leftrightarrow \varphi_2) \Leftrightarrow (\varphi_2 \Leftrightarrow \varphi_1)}\ \textsf{rewrite\_iff\_symm}$$

$$\frac{}{(\neg\top) \Leftrightarrow \bot}\ \textsf{rewrite\_not\_true} \qquad \frac{}{(\neg\bot) \Leftrightarrow \top}\ \textsf{rewrite\_not\_false}$$

$$\frac{}{(\varphi \Leftrightarrow \top) \Leftrightarrow \varphi}\ \textsf{rewrite\_iff}_0 \qquad \frac{}{(\varphi \Leftrightarrow \bot) \Leftrightarrow \neg\varphi}\ \textsf{rewrite\_iff}_1$$

$$\frac{}{(\neg\varphi \Leftrightarrow \varphi) \Leftrightarrow \bot}\ \textsf{rewrite\_iff}_2 \qquad \frac{\varphi_1 \wedge \ldots \wedge \varphi_n}{\varphi_i}\ \textsf{andE}$$

$$\frac{\varphi_1 \Leftrightarrow \varphi_2 \quad \varphi_3 \Leftrightarrow \varphi_4}{\varphi_1 \Box\ \varphi_3 \Leftrightarrow \varphi_2 \Box\ \varphi_4}\ \textsf{basic\_subst\_op}_2$$

where $\Box \in \{\wedge,\ \vee,\ \Leftrightarrow\}$.

A.4 Equality Rules

$$\frac{}{t_1 = t_1}\ \textsf{refl}$$

$$\frac{t_1 = t_2 \quad t_3 = t_4}{t_1 \sim t_3 \Leftrightarrow t_2 \sim t_4}\ \textsf{basic\_subst\_op}_1 \qquad \frac{t_1 = t_2 \quad t_2 = t_3}{t_1 \ = t_3}\ \textsf{eq\_trans}$$

$$\frac{t_1 = t_2 \quad t_3 = t_4}{t_1 \bowtie t_3 = t_2 \bowtie t_4}\ \textsf{basic\_subst\_op1} \qquad \frac{t_1 = t_2}{t_2 = t_1}\ \textsf{eq\_symm}$$

A.5 Theory Rules

$$\frac{t_1 \prec_1 t_2 \quad t_2 \prec_2 t_3}{t_1 (\prec_1 \cdot \prec_2)\ t_3}\ \textsf{real\_shadow} \qquad \frac{t_1 \leq t_2 \quad t_2 \leq t_1}{t_1 = t_2}\ \textsf{real\_shadow\_eq}$$

$$\frac{t_1 \prec_1 t_2 \quad t_3 \prec_2 t_4}{t_1 + t_3 (\prec_1 \cdot \prec_2)\ t_2 + t_4}\ \textsf{add\_inequalities} \qquad \frac{t_1 = s_1}{-t_1 = -s_1}\ \textsf{basic\_subst\_op0}_1$$

$$\frac{}{-t = (-1) \cdot t}\ \textsf{uminus\_to\_mult} \qquad \frac{}{ite(\varphi, t, t) = t}\ \textsf{rewrite\_ite\_same}$$

$$\frac{}{(t_1 - t_2) = t_1 + (-1 \cdot t_2)}\ \textsf{minus\_to\_plus} \qquad \frac{\{t' \text{ canonical form of } t\}}{t = t'}\ \textsf{canon}$$

$$\frac{t_1 \leq t_2 + c_1 \quad t_2 \leq t_1 + c_2 \quad \{c_1 + c_2 = 0\}}{t_1 = t_2 + c_1 \wedge t_2 = t_1 + c_2} \ \textsf{implyEqualities}$$

A.6 CNF Conversion Rules

$$\frac{}{(\varphi_1 \Rightarrow \varphi_2) \vee \varphi_1} \ \textsf{CNF\_imp\_0} \qquad \frac{}{(\varphi_1 \Rightarrow \varphi_2) \vee \neg\varphi_2} \ \textsf{CNF\_imp\_1}$$

$$\frac{}{\neg(\varphi_1 \Rightarrow \varphi_2) \vee \neg\varphi_1 \vee \varphi_2} \ \textsf{CNF\_imp\_2}$$

$$\frac{}{(\varphi_1 \Leftrightarrow \varphi_2) \vee \varphi_1 \vee \varphi_2} \ \textsf{CNF\_iff\_0} \qquad \frac{}{(\varphi_1 \Leftrightarrow \varphi_2) \vee \neg\varphi_1 \vee \neg\varphi_2} \ \textsf{CNF\_iff\_1}$$

$$\frac{}{\neg(\varphi_1 \Leftrightarrow \varphi_2) \vee \neg\varphi_1 \vee \varphi_2} \ \textsf{CNF\_iff\_2} \qquad \frac{}{\neg(\varphi_1 \Leftrightarrow \varphi_2) \vee \varphi_1 \vee \varphi_2} \ \textsf{CNF\_iff\_3}$$

$$\frac{}{\neg(\varphi_1 \wedge \ldots \wedge \varphi_n) \vee \varphi_i} \ \textsf{CNF\_and\_mid} \qquad \frac{}{(\varphi_1 \wedge \ldots \wedge \varphi_n) \vee \neg\varphi_1 \vee \ldots \vee \neg\varphi_n} \ \textsf{CNF\_and\_final}$$

$$\frac{}{(\varphi_1 \vee \ldots \vee \varphi_n) \vee \neg\varphi_i} \ \textsf{CNF\_or\_mid} \qquad \frac{}{\neg(\varphi_1 \vee \ldots \vee \varphi_n) \vee \varphi_1 \vee \ldots \vee \varphi_n} \ \textsf{CNF\_or\_final}$$

$$\frac{}{\neg ite(\phi, \varphi_1, \varphi_2) \vee \phi \vee \varphi_2} \ \textsf{CNFITE\_0} \qquad \frac{}{ite(\phi, \varphi_1, \varphi_2) \vee \phi \vee \neg\varphi_2} \ \textsf{CNFITE\_1}$$

$$\frac{}{ite(\phi, \varphi_1, \varphi_2) \vee \neg\phi \vee \neg\varphi_1} \ \textsf{CNFITE\_2} \qquad \frac{}{\neg ite(\phi, \varphi_1, \varphi_2) \vee \neg\phi \vee \varphi_1} \ \textsf{CNFITE\_3}$$

$$\frac{}{ite(\phi, \varphi_1, \varphi_2) \vee \phi \vee \neg\varphi_1 \vee \neg\varphi_2} \ \textsf{CNFITE\_4} \qquad \frac{}{\neg ite(\phi, \varphi_1, \varphi_2) \vee \varphi_1 \vee \varphi_2} \ \textsf{CNFITE\_5}$$

A.7 Coarse Grained Rules

$$\frac{\{\varphi' \text{ canonical form of } \varphi\}}{\varphi \Leftrightarrow \varphi'} \ \textsf{rewrite\_and, rewrite\_or}$$

$$\frac{P_1 : t_1 \prec_1 t_1' \quad \ldots \quad P_n : t_n \prec_n t_n'}{\bot} \ \textsf{cycleConflict}$$

where $(t_1 \prec_1 t_1') \wedge \ldots \wedge (t_n \prec_n t_n') \Rightarrow \bot$.

$$\frac{t_1 = s_1 \quad \ldots \quad t_n = s_n}{t = t[s_1/t_1, \ldots, s_n/t_n]} \ \textsf{optimized\_subst\_op}_1$$

$$\frac{\varphi_1 \Leftrightarrow \psi_1 \quad \ldots \quad \varphi_n \Leftrightarrow \psi_n}{\varphi \Leftrightarrow \varphi[\psi_1/\varphi_1, \ldots, \psi_n/\varphi_n]} \ \textsf{optimized\_subst\_op}_2$$

B $\mathcal{L}$ Specific Proof Rules

B.1 Axiom Rules

$$\frac{}{0 = 0}\ \mathsf{lra\_axiom=} \qquad \frac{\{c > 0\}}{c > 0}\ \mathsf{lra\_axiom>}$$

$$\frac{\{c \geq 0\}}{c \geq 0}\ \mathsf{lra\_axiom\geq} \qquad \frac{\{c \neq 0\}}{c \neq 0}\ \mathsf{lra\_axiom\neq}$$

B.2 Equality Deduction Rule

$$\frac{p \geq 0 \quad p' \geq 0 \quad \{p + p' = 0\}}{p = 0}\ \mathsf{lra{\geq}{\geq}to{=}}$$

B.3 Contradiction Rules

$$\frac{p = 0 \quad \{p \neq 0\}}{\bot}\ \mathsf{lra\_contra}_{=} \qquad \frac{p > 0 \quad \{p \not> 0\}}{\bot}\ \mathsf{lra\_contra}_{>}$$

$$\frac{p \geq 0 \quad \{p < 0\}}{\bot}\ \mathsf{lra\_contra}_{\geq} \qquad \frac{p \neq 0 \quad \{p = 0\}}{\bot}\ \mathsf{lra\_contra}_{\neq}$$

B.4 Multiplication Rules

$$\frac{p = 0}{(c \cdot p){\downarrow} = 0}\ \mathsf{lra\_mult\_}c_{=} \qquad \frac{p > 0 \quad \{c > 0\}}{(c \cdot p){\downarrow} > 0}\ \mathsf{lra\_mult\_}c_{>}$$

$$\frac{p \geq 0 \quad \{c \geq 0\}}{(c \cdot p){\downarrow} \geq 0}\ \mathsf{lra\_mult\_}c_{\geq} \qquad \frac{p \neq 0 \quad \{c \neq 0\}}{(c \cdot p){\downarrow} \neq 0}\ \mathsf{lra\_mult\_}c_{\neq}$$

B.5 Addition Rules

$$\frac{p_1 = 0 \quad p_2 = 0}{(p_1 + p_2){\downarrow} = 0}\ \mathsf{lra\_add}_{==} \qquad \frac{p_1 > 0 \quad p_2 > 0}{(p_1 + p_2){\downarrow} > 0}\ \mathsf{lra\_add}_{>>}$$

$$\frac{p_1 \geq 0 \quad p_2 \geq 0}{(p_1 + p_2){\downarrow} \geq 0}\ \mathsf{lra\_add}_{\geq\geq} \qquad \frac{p_1 = 0 \quad p_2 > 0}{(p_1 + p_2){\downarrow} > 0}\ \mathsf{lra\_add}_{=>}$$

$$\frac{p_1 = 0 \quad p_2 \geq 0}{(p_1 + p_2){\downarrow} \geq 0}\ \mathsf{lra\_add}_{=\geq} \qquad \frac{p_1 > 0 \quad p_2 \geq 0}{(p_1 + p_2){\downarrow} > 0}\ \mathsf{lra\_add}_{>\geq}$$

$$\frac{p_1 = 0 \quad p_2 \neq 0}{(p_1 + p_2){\downarrow} \neq 0}\ \mathsf{lra\_add}_{=\neq}$$

B.6 Subtraction Rules

$$\frac{p_1 = 0 \quad p_2 = 0}{(p_1 - p_2){\downarrow} = 0}\ \mathsf{lra\_sub}_{==} \qquad \frac{p_1 > 0 \quad p_2 = 0}{(p_1 - p_2){\downarrow} > 0}\ \mathsf{lra\_sub}_{>=}$$

$$\frac{p_1 \geq 0 \quad p_2 = 0}{(p_1 - p_2){\downarrow} \geq 0}\ \mathsf{lra\_sub}_{\geq=} \qquad \frac{p_1 \neq 0 \quad p_2 = 0}{(p_1 - p_2){\downarrow} \neq 0}\ \mathsf{lra\_sub}_{\neq=}$$

B.7 Term Normalization Rules

In the rules below c_{t} and c_{p} denote the same rational constant, in one case considered of term type and in the other as of polynomial type (similarly for the variables v_{t} and v_{p}).

$$\frac{}{c_{\mathrm{t}} = c_{\mathrm{p}}}\ \mathsf{poly\_norm\_const} \qquad \frac{t_1 = p_1 \quad t_2 = p_2}{t_1 + t_2 = (p_1 + p_2){\downarrow}}\ \mathsf{poly\_norm}_{+}$$

$$\frac{}{v_{\mathrm{t}} = v_{\mathrm{p}}}\ \mathsf{poly\_norm\_var} \qquad \frac{t_1 = p_1 \quad t_2 = p_2}{t_1 - t_2 = (p_1 - p_2){\downarrow}}\ \mathsf{poly\_norm}_{-}$$

$$\frac{t = p}{c_{\mathrm{t}} \cdot t = (c_{\mathrm{p}} \cdot p){\downarrow}}\ \mathsf{poly\_norm}_{c\cdot} \qquad \frac{t = p}{t \cdot c_{\mathrm{t}} = (p \cdot c_{\mathrm{p}}){\downarrow}}\ \mathsf{poly\_norm}_{\cdot c}$$

$$\frac{t = p}{-t = (-p){\downarrow}}\ \mathsf{poly\_norm}_{u-}$$

The following rules are used to normalize *ite* terms as polynomials:

$$\frac{\begin{matrix}[t = v] \\ \vdots \\ \varphi\end{matrix}}{\varphi}\ \mathsf{atomize\_term} \qquad \frac{t = v}{t = v_p}\ \mathsf{poly\_norm\_atom}$$

where v is introduced as a fresh variable in the atomize_term rule.

B.8 Equation Normalization Rules

$$\frac{t_1 = t_2 \quad t_1 - t_2 = p}{p = 0}\ \mathsf{poly\_norm}_{=} \qquad \frac{t_1 \neq t_2 \quad t_2 - t_1 = p}{p \neq 0}\ \mathsf{poly\_norm}_{\neq}$$

$$\frac{t_1 > t_2 \quad t_1 - t_2 = p}{p > 0}\ \mathsf{poly\_norm}_{>} \qquad \frac{t_1 < t_2 \quad t_2 - t_1 = p}{p > 0}\ \mathsf{poly\_norm}_{<}$$

$$\frac{t_1 \geq t_2 \quad t_1 - t_2 = p}{p \geq 0}\ \mathsf{poly\_norm}_{\geq} \qquad \frac{t_1 \leq t_2 \quad t_2 - t_1 = p}{p \geq 0}\ \mathsf{poly\_norm}_{\leq}$$