Available online at www.sciencedirect.com
ScienceDirect
Electronic Notes in Theoretical Computer Science
ELSEVIER
Electronic Notes in Theoretical Computer Science 212 (2008) 163–176
www.elsevier.com/locate/entcs

Rewriting Interpolants

Christopher Lynch [1]

Department of Mathematics and Computer Science
Clarkson University
Potsdam, USA

Yuefeng Tang[2]

Department of Mathematics and Computer Science
Clarkson University
Potsdam, USA

Abstract

We give a method of constructing an interpolant for linear equality, and inequality constraints over the rational numbers. Our method is based on efficient rewriting techniques, and does not require the use of combination methods. The interpolant is constructed in such a way that it reflects the structure of the rewrite proof.

Keywords: interpolants, rewriting

1 Introduction

Given two logical formulas A and B, a formula P is an interpolant of the pair (A, B) if P is implied by A, P implies B and P contains only symbols common to A and B. The idea of interpolant was defined by Craig in 1957 [1]. Basically, an interpolant defines an interface through which A and B interact, using only information that is shared between the two formulas. Recently, McMillan has shown that interpolants can be useful for verification, for finding inductive invariants, and for predicate abstraction [3,4,5].

For example, in model checking, suppose we want to find the inductive invariant of a process. We have some initial constraints(precondition), some transition constraints, and some final constraints (postcondition) that we would like to be true. We set A as the initial constraints plus potentially some transition constraints, and

[1] Email: clynch@clarkson.edu
[2] Email: tangy@clarkson.edu

1571-0661 © 2008 Elsevier B.V. Open access under CC BY-NC-ND license.
doi:10.1016/j.entcs.2008.04.060

B as the final constraints plus potentially some transition constraints. We need to prove that A implies B, but to prove this in general for any possible number of transitions, we need to find an invariant that will be true for any possible number of transitions. Since an interpolant P represents some formula "in between" A and B, there is a possibility that an interpolant may be an invariant, or that there could be a fixed point operation to begin with this interpolant and develop an invariant.

In refutational theorem proving, we find a proof of the unsatisfiability of $A \wedge \neg B$. Therefore, most definitions of interpolation nowadays say that P is a formula such that $A \models P$, P is inconsistent with B, and P only contains symbols shared by A and B. One way to find an interpolant is to examine the refutation proof of $A \wedge \neg B$ and construct an interpolant from the proof. We augment each constraint in the proof with an "interpolant part". We prove that this augmentation has certain properties, that are preserved by the inferences, and that will imply that the "interpolant part" of the contradiction $\perp$ is actually an interpolant for the proof.

In this paper, we present a method for such augmentations in refutation proofs of linear constraints over the rationals. However, because our approach is based on rewriting, we can easily extend our method to handle dis-equalities and uninterpreted function symbols without the combination of different methods. We show the correctness of the augmentations, and the soundness and completeness of the inference system. However, we believe we have developed something more general. We consider it to be a useful framework to provide mappings and develop interpolants for other inference systems.

There can be an infinite number of interpolants for pairs of formulas (A, B). Our interest is to develop the interpolant in such a way as to be a representation of the way inferences are performed in the proof. To this end, we use substitutions to represent sharing between a constraint and its interpolant part. We represent the sharing with a shared variable between them, and a substitution part to store the expression that belongs there. We found that substitutions are also necessary to ensure that our inference steps preserve the required properties for guaranteeing an interpolant at the end of the proof. We believe our framework is quite general and can apply to other inference systems. We also think it is a good way to try to mimic the steps of the proof in the interpolant.

We first present our new inference system without interpolants, but we present it in such a way that interpolants can be introduced later. We prove the completeness of this inference system. Then we add interpolant parts and substitutions to these terms, and present the inference rules in this framework. We prove that each inference rule preserves the required properties for guaranteeing an interpolant in the end. This inference system is complete, because without the substitutions and interpolants, it is the same inference system as given earlier. We conclude with related and future work.

2 Preliminaries

We assume a signature Σ containing constants. A *term* is defined as a rational number, a constant, or kx where k is a rational number and x is a constant. In this context k is also called the coefficient of the constant x. An *expression* is a term or a summation of terms. $t_1 - kt_2$ is an abbreviation for $t_1 + (-k)t_2$ where t_1 and t_2 are terms. The symbol $+$ is a predefined function symbol, which is not in Σ, which will be given the usual meaning. Instead of considering it as an associative and commutative binary symbol, we just consider it as a symbol applied to a multiset of terms.

Terms are always assumed to be *simplified*. This means that $0x$ is reduced to zero. Also, expressions are *simplified*. A simplified expression is a simplified term, or a summation of non-zero simplified terms where no constant appears twice an no numeric term appears twice.

A constraint is $\top$ or $\bot$ or of the form $s\phi t$, where s and t are expressions and $\phi \in \{\geq, \leq, =, <, >\}$. Given an operator ϕ, we define an operator ϕ^n. ϕ^n is the nonstrict version of ϕ, defined as follows. If $\phi \in \{=\}$ then ϕ^n is '='. If $\phi \in \{<, \leq\}$ then ϕ^n is '$\leq$'. Also, we say that '$\geq$' and '$>$' have the *opposite direction* of '$\leq$' and '$<$'.

Operators have their usual meaning in arithmetic. We say that a constraint (set of constraints) is *satisfiable* if the constants can be substituted by rational numbers to make the constraint (set of constraints) true. A constraint (set of constraints) is *unsatisfiable* if it is not satisfiable.

In the paper, we are going to consider two sets of constraints: A and B. Let Σ_A be the signature of A and Σ_B be the signature of B. Then $\Sigma = \Sigma_A \cup \Sigma_B$. Σ_A and Σ_B are not necessarily disjoint. Given A and B, elements of $\Sigma_A \cap \Sigma_B$ are *global*. Elements which only exist in Σ_A or Σ_B are *local*. A term (expression) is *global* if all its symbols are global otherwise it's local. If a local term (expression) contains no B-local symbols, then it is A-local. If a local term(expression) contains no A-local symbols, then it is B-local.

Given a set of constraints A and a conjunction of constraints p, we write $A \models p$ to mean that p is true whenever A is true. Given sets of constraints A and B such that $A \cup B$ is unsatisfiable, we define a conjunction of constraints p to be an *interpolant* of (A, B) if $A \models p$, $B \cup \{p\}$ is unsatisfiable and p contains only global variables.

We assume an ordering $\prec$ on constants, which is well-founded, and total. We require that the local constants are larger than global constants. We extend the ordering to a well-founded total ordering on terms, where $k_1 x \prec k_2 y$ if $x \prec y$ for rational numbers k_1 and k_2 and constants x and y. The expression $t_1 + \cdots + t_n$ is viewed as a multiset $\{t_1, \cdots, t_n\}$. Operators are ordered with the following precedence: $\{=\} \prec \{\leq\} = \{\geq\} \prec \{<\} = \{>\}$. Then a constraint $s\phi t$ is given the measure (ϕ, s, t), and constraints are ordered by comparing their measures lexicographically.

In the next section, we assume that constraints are *simplified*, so that they are

always of the form $x\phi t$ where x is a constant larger than every constant in t, and t is simplified. Tautologies are simplified to $\top$. Contradictions are simplified to $\bot$. Notice that if a simplified constraint is unsatisfiable then it must not contain any constants from Σ.

Let SV be a set of *substitution variables* disjoint from Σ. A *substitution* is a mapping from a substitution variable to an expression, which is homomorphically extended to all terms, expressions and constraints. We consider a substitution $\{V \mapsto t\}$ to be *simplified* if t is simplified, and the largest term in t has a coefficient of 1 unless t is 0. We assume substitutions are always simplified, and we always write them in the form $\{V \mapsto x + t\}$ where x is larger than any constant in t.

The formulas used in our proof will not be just constraints, but they will be triples $c|i|\theta$, where c is a constraint, i is a conjunction of constraints, and θ is a substitution. We will call i the *interpolant part* of c to emphasize that it will lead to an interpolant. We are going to give an inference system so that whenever $\bot|i|\theta$ is constructed then $i\theta$ is an interpolant.

For example, consider the triple $V = 0|V = y \wedge z = 0|\{V \mapsto x\}$, where c is $V = 0$, i is $V = y \wedge z = 0$ and θ is $\{V \mapsto x\}$. In this case x, y and z are constants, and V is a substitution variable. This is really intended to represent the constraint $c\theta$, which is $x = 0$, but x has been abstracted out of the substitution part so it can be shared with $i\theta$, which is $x = y \wedge z = 0$, which will be used to help us construct an interpolant.

To simplify matters, we first give our refutational inference rules with single constraints. Then we show how to extend that inference system to triples, in order to find interpolants.

3 Inference Rules

In this section we present an inference system for determining the unsatisfiability of a set of constraints. Inferences will be applied to a set of constraints. After defining the inference system, we will prove that $\bot$ can be generated by these inference rules from a set S of constraints if and only if S is unsatisfiable.

This inference system only involves single constraints. In later sections, we will show how to use this inference system to determine interpolants. At that point, we will need to extend this inference system using triples instead of single constraints.

We are assuming that all our constraints are simplified. After each inference, the conclusion is converted into simplified form.

There is only one inference rule **Rewriting Rule**. **Rewriting Rule** replaces a big element in a formula by a smaller one:

$$\frac{x\phi_1 t_1 \quad x\phi_2 t_2}{t_1 \ \phi_2 \ t_2} \tag{1}$$

Since the constraints are simplified, x must be larger than all constants in expressions t_1 and t_2. ϕ_1 and $\phi_2 \in \{=, \geq, \leq, >, <\}$. We always assume the right premise is larger than or equal to the left one. If neither premise is an equation, the direction of ϕ_1 and ϕ_2 must be opposite. Otherwise this inference rule is not

applicable.

After applying the Rewriting Rule, the constant x in the right premise is substituted by t_1. The conclusion is smaller than the right premise.

Next we give an example. Given two sets $A = \{x = y, x \geq 3z\}$, $B = \{y < z, z = 1\}$, suppose the ordering of the constants is $z \prec y \prec x$ [3]. Here is a process to derive a contradiction using the Rewriting Rule.

Example 3.1

$$\dfrac{\dfrac{\dfrac{x = y \quad x \geq 3z}{y \geq 3z} \quad y < z}{z < 0} \quad z = 1}{\bot} \quad z < 0$$

The Conclusion has been simplified after each inference rule. This proves that $A \cup B$ is inconsistent.

4 Completeness Proof

In this section, we prove the completeness of the inference rules defined in the previous section. In other words, we show that it is always possible to derive $\bot$ from an unsatisfiable set of constraints. First we need some definitions to model a derivation.

Definition 4.1 Given a constraint c and a set of constraints S, c is *redundant* in S if there exist $c_1, \cdots, c_n \in S$ such that $c_1, \cdots, c_n \models c$ and $c_i \prec c$ for all i. A set of constraints S is *saturated* if the conclusion of every inference of members of S is in S or is redundant in S.

A saturated set represents the result of an exhaustive application of our inference rules on a set of clauses. So in this section, we will deal with saturated sets. For completeness, we need to prove that if a saturated set S is unsatisfiable, then $\bot \in S$. Since the inference rules are sound, every inference preserves (un)satisfiability.

To simplify the discussion in this section, we will define sequences such as $t_0 < t_1 \leq t_2$ to be called paths.

Definition 4.2 A *ϕ path over S proving $t_0 \phi t_n$*, where $\phi \in \{<, >, \leq, \geq, =\}$, is a sequence $t_0, t_1, \cdots, t_n$, such that $n > 0$ and for each $i > 1$ there is a constraint $s_{i-1}\phi_i s_i$ in S, an expression e_i and a number k_i such that t_i is the same as $k_i s_i + e_i$, and t_{i-1} is the same as $k_i s_{i-1} + e_i$. If ϕ is '$\geq$' then all ϕ_i must be '$\geq$','$>$', or '$=$'. If ϕ is '$>$' then all ϕ_i must be '$\geq$', '$=$' or '$>$', and at least one ϕ_i must be '$>$'. Analogously for when ϕ is '$\leq$' or '$<$'. If ϕ is '$=$' then all ϕ_i are '$=$'.

Notice that if $S \models t_0 \geq t_n$ then there must be a $\geq$ path over S proving $t_0 \geq t_n + k$ for some non-negative number k. If $S \models t_0 > t_n$ then

[3] x must be the largest constant since it is local.

(i) there is a $>$ path over S proving $t_0 > t_n + k$ for some non-negative number k or

(ii) there is a $\geq$ path over S proving $t_0 \geq t_n + k$ for some positive number k.

If $S \models t_0 = t_n$ then there is a $\geq$ path over S proving $t_0 \geq t_n$ and a $\geq$ path over S proving $t_n \geq t_0$, and there might be an '=' path over S proving $t_0 = t_n$.

We define a ϕ path over S proving $s\phi t$ to be a ϕ *smallest path over* S *proving* $s\phi t$ if there is no smaller ϕ path over S proving $s\phi t$. The paths are compared by considering them as multisets, and using the multiset extension of $\prec$.

We say that a ϕ path proving $t_0\phi t_n$ contains a *peak* if there are consecutive terms t_{i-1}, t_i and t_{i+1} in the path such that $t_{i-1} \prec t_i$ and $t_{i+1} \prec t_i$. If we assume $t_0 \succ t_n$, then the lack of a peak implies that t_0 is the largest term on the path.

Lemma 4.3 *Let S be a saturated set. Let P be a smallest path proving $t_0\phi t_n$ over S. Then P does not contain a peak.*

Proof. Assume P is a smallest path containing a peak proving $t_0\phi t_n$. So there is an i such that $t_i \succ t_{i-1}$ and $t_i \succ t_{i+1}$. By definition of path there are corresponding constraints $s_{i-1}\phi_i s_i$ and $u_i\phi_{i+1}u_{i+1}$. Since $t_i \succ t_{i-1}$ and $t_i \succ t_{i+1}$, we know that $s_i \succ s_{i-1}$ and $u_i \succ u_{i+1}$. If s_i and u_i occur at disjoint positions in t_i then we can first apply $u_i\phi_{i+1}u_{i+1}$ to t_{i-1} and then $s_{i-1}\phi_i s_i$ to get a shorter proof. So assume that s_i and u_i occur at overlapping positions in t_i. Then there is an inference between $s_{i-1}\phi_i s_i$ and $u_i\phi_{i+1}u_{i+1}$ which produces $s_{i-1}\phi' u_{i+1}$. This constraint is either in S or is redundant in S. If it is in S, we can remove t_i from the path and get a smaller path. If it is redundant in S, then there are smaller constraints in S which imply $s_{i-1}\phi' u_{i+1}$. Therefore there must be a ϕ' path proving $s_{i-1}\phi' u_{i+1}$ which uses only terms smaller than s_i. In that case, t_i can be removed from its path and replaced by smaller terms. This contradicts the fact that the original path is a smallest path.□

The lemma will be used to prove the completeness theorem. We need to prove that in an unsatisfiable set of constraints, we can always generate $\perp$. We will prove this by showing that if we saturate a set of constraints and do not generate the empty clause then we can build a model, i.e., a consistent set of constraints that implies all the constraints in the saturated set, as in [6].

Theorem 4.4 *Let S be a saturated set of constraints. If S is unsatisfiable then $\perp \in S$.*

Proof. We prove the contrapositive. Suppose that $\perp \notin S$. We will create a set M such that $M \models S$, proving that S is satisfiable. Let $S = \{c_1, \cdots, c_n\}$. We give the following co-inductive definitions of M_i and P_i for each constraint c_i in S. They are defined as follows. Let $M_i = \bigcup_{c_j \prec c_i} P_j$. Let $P_i = \{c_i\}$ if $M_i \not\models c_i$. Otherwise, $P_i = \emptyset$. If $P_i = \{c_i\}$, we say that c_i is productive. Let $\mathrm{M} = \bigcup_{c_i \in S} P_i$. We need to prove that M is a consistent set.

To prove that M is consistent, we consider each c_i in S, and prove by induction on the size of c_i that if c_i is productive then $M_i \cup c_i$ is consistent.

The constraint c_i is of the form $s\phi t$. We will assume that ϕ is the constraint $<$, since all other cases are similar.

By the form of the constraints, we know that $s \succ t$. If $M_i \cup \{s < t\}$ is inconsistent, then $M_i \models s \geq t$. Therefore there must be a $\geq$ path proving $s \geq t + k$, for some nonnegative number k, in S.

We consider the smallest such path. This path cannot contain a peak. So the first step on this path must be of the form $s\phi t_1$, where $t_1 \prec s$, and ϕ is $\geq$, $>$ or $=$. This means that $s\phi t_1 \in M_i$. Also, it must be the case that $M_i \models t_1 \geq t$. There is an inference between $s < t$ and $s\phi t_1$, whose result is $t > t_1$. By the saturation of S, this means that $M_i \models t > t_1$. This conflicts with the fact that $M_i \models t_1 \geq t$. Therefore, $M_i \cup \{s < t\}$ must be consistent.

□

5 Adding Interpolant Parts

In the previous part, we only defined the inference rules without considering the interpolant. In this section we will introduce the method to obtain the interpolant based on the inference rules.

We must show how to take the original constraints from A and B and turn them into triples. In addition, we need to show how the inference rules apply to those triples. The plan is the following.

(i) Give inference rules to take a constraint c' from A or B and convert it into a triple $c|i|\theta$. This triple must have the property that c' is $c\theta$, since we are going to simulate the inference system given earlier.

(ii) Give inference rules to simulate the inference rules already given, but using triples this time. Every inference rule given earlier must be simulated, and if we instantiate the conclusion constraint with the conclusion substitution then we must obtain the same conclusion as in the earlier inference rules.

(iii) We finally, need to prove that $i\theta$ is an interpolant of (A, B) when c becomes $\bot$. We will prove this by showing an incremental fact. We will show that for every triple $c|i|\theta$, $A \models i\theta$ and $B \wedge i \models c$. Those two properties are sufficient to ensure that $i\theta$ will be an interpolant when c becomes $\bot$, along with the fact that $i\theta$ is global.

We will try to give more motivation for the use of the substitution. The reason for the substitution is in order to share structure between c and i. The effect of our inference rules will be that inferences involving A will take place in a substitution, thereby allowing those inferences to involve i in addition to c. However, inferences involving B will not take place inside a substitution, thus we will force the inference to involve c but not i. This will ensure that, as much as possible, inferences involving A are recorded in i but not inferences involving B. Therefore, we will keep a trace of the A-inferences performed. This allows the interpolant to be formed the same way as the inferences were executed. We believe this is the interpolant which is often needed in practice. There are infinitely many interpolants for any given unsatisfiable

pair (A, B) but we believe this is the interpolant that is really recording the way the proof is being done, and it will be more useful in applications, such as finding loop invariants.

We will define a triple to be correct if it is in the form we mentioned above. Additionally, we also need a condition about the global and local constraints in order to ensure that a triple containing the constraint $\perp$ only has global variables in the interpolant part.

Definition 5.1 Let A and B be sets of constraints. A triple $c|i|\theta$ is *geographically correct* if

(i) when $i\theta$ is simplified, it is not B-local.

(ii) if $i\theta$ is A-local, then i is identical to c

(iii) if i is identical to c, then $c\theta$ is implied by A, and

(iv) θ does not contain both A-local and B-local constants.

Definition 5.2 Let A and B be sets of constraints. A triple $c|i|\theta$ is (A, B)-*correct* if

(i) $A \models i\theta$,

(ii) $B \wedge i \models c$, and

(iii) $c|i|\theta$ is geographically correct

Based on the way that we build triples, a triple always has the format $V\phi 0|V\phi 0|\theta$ or $V\phi 0|V\phi t \wedge i'|\theta$ where i' does not contain substitution variables.

We begin by defining inference rules for creating triples from initial constraints depending on whether the initial constraint is from A or B. We call this rule **Make-Triple**:

$$\frac{x\phi t}{V\phi 0|V\phi 0|\{V \mapsto x - t\}}$$

where $x\phi t \in A$.

$$\frac{x\phi t}{V\phi 0|V\phi^n x - t|\{V \mapsto x - t\}}$$

where $x\phi t \in B$, and ϕ^n defined in the preliminaries is the nonstrict version of ϕ.

Note that in constraints from A, we use new the substitution variable V to share everything of c in the substitution part i, because we want to record all the actions involving A in the interpolant. However, in constraints from B, we also use V to share c with i but we relate that to an original copy of the constraint which will not be modified by the inferences. This allows us to record the actions that occur on that constraint, not the actions using that constraint.

Lemma 5.3 *If c is a constraint from $A \cup B$, then the result of applying Make-Triple to c is (A, B)-correct.*

Proof. Let's look at the first Make-Triple rule.

(i) Since $x\phi t$ is from A, the first property $A \models x - t\phi 0$ is true.

(ii) The second property $B \wedge V\phi 0 \models V\phi 0$ is obviously true.

(iii) Next is to show the third property that the conclusion is geographically correct.
  (a) $(V\phi 0)\theta$ is not B-local where θ is $\{V \mapsto x - t\}$ because the constraint $x\phi t$ is from A.
  (b) The interpolant part is identical to its constraint. So this property is true.
  (c) Since the constraint is from A, $(V\phi 0)\theta$ must be implied by A.
  (d) θ is A-local or global.

Next, let's look at the second Make-Triple rule.

(i) $(V\phi^n x - t)\theta$ is a tautology. So, the first property $A \models (V\phi^n x - t)\theta$ is true.

(ii) Let's suppose ϕ is $\geq$ then ϕ^n is $\geq$ too. The proof for other operators is similar to the $\geq$ case. So, $x \geq t$ from B along with $V \geq x - t$ implies $V \geq 0$. Thus, the second property $B \wedge V \geq x - t \models V \geq 0$.

(iii) Next is to show the third property that the conclusion is geographically correct.
  (a) since $(V\phi^n x - t)\theta$ is a tautology, $i\theta$ when simplified is not B-local.
  (b) The constraint is from B. So, this property is true.
  (c) The interpolant part is not identical to its constraint. So, this property is true.
  (d) θ could be B-local or *global*.

Therefore, the result of applying Make-Triple to c is (A, B)-correct.

□

As we stated in the previous section, we have to simplify each newly generated constraint after applying an inference rule. Given an expression or a constraint, it is well-known how to simplify it. But after an inference rule, the substitution part might not be simplified anymore, because the coefficient of the largest term may not be positive one. Here we give an inference rule called **Simplify Rule** to simplify the substitution part of a constraint. This will be performed eagerly when each new constraint is generated.

$$\frac{V\phi_1 0 | V\phi_2 s \wedge i' | \{V \mapsto kx + t\}}{kW\phi_1 0 | kW\phi_2 s \wedge i' | \{W \mapsto x + t/k\}}$$

After this, it is simplified in the obvious way.

Example 5.4

$$\frac{V_3 < 0 | V_3 \leq y - z | \{V_3 \mapsto 2z\}}{2V_5 < 0 | 2V_5 \leq y - z | \{V_5 \mapsto z\}}$$

Then, the conclusion can be simplified to the triple $V_5 < 0 | V_5 \leq \frac{1}{2}z - \frac{1}{2}y | \{V_5 \mapsto z\}$

Lemma 5.5 *An (A, B)-correct triple always simplifies to an (A, B)-correct triple.*

Proof. All properties of (A, B)-correct are trivial except the second property. But, the second property follows the fact that i' does not contain the substitution variable. □

Next, we will show the redefined inference rules. The newly defined inference rules will be strictly based on the inference rule defined in the previous part, but

the inference will take place in the substitution, and we will control when sharing is allowed and when it is not allowed.

We use two inference rules **Triple-Rewriting Rule** and **Combination Rule** to simulate the **Rewriting Rule** defined in the previous part. For ease of presentation, before applying these rules, we switch positions of the two premises if the left premise's interpolant part is not identical to its constraint and the right premise is non-strict inequality whose interpolant part is identical to its constraint. Then, if two premises have the format defined in the following Triple-Rewriting Rule, we apply the Triple-Rewriting Rule. Otherwise, we apply the Combination Rule.

The **Triple-Rewriting Rule** is the following:

$$\frac{V_1\phi_1 0|V_1\phi_1 0|\{V_1 \mapsto x - t_1\} \quad V_2\phi_2 0 \ |i|\{V_2 \mapsto x - t_2\}}{V_2\phi_2 0|i|\{V_2 \mapsto t_1 - t_2\}}$$

Notice that the left premise is implied by A based on the format of the triple and the conclusion looks like the right premise, except that the substitution has changed, thereby recording the inference into the eventual interpolant.

The Triple Rewriting Rule preserves correctness:

Lemma 5.6 *If the premises of a Triple-Rewriting Rule are (A, B)-correct triples, then the conclusion is too.*

Proof. Let θ_1 be $\{x - t_1\}$, θ_2 be $\{x - t_2\}$ and θ_2' be $\{t_1 - t_2\}$.

(i) Let's assume ϕ_1 is $\geq$, and ϕ_2 is $\leq$. Proofs for other cases of the Triple-Rewriting Rule are similar to this one. Suppose i has the format $V_2 \leq t \wedge i'$. Our goal is to prove the first property $A \models (t_1 - t_2 \leq t) \wedge i'$. Based on the assumption, $A \models x - t_1 \geq 0$ and $A \models (x - t_2 \leq t) \wedge i'$. Then, we can easily conclude $t_1 - t_2 \leq t$ from $x - t_1 \geq 0$ and $x - t_2 \leq t$. Thus, the first property is true.

(ii) The second property is obviously true because $V_2\phi 0$ and i remain the same as the second premise.

(iii) Next is to show the conclusion is geographically correct.
   (a) since the interpolant part $V_1\phi_1 0$ is identical to the constraint in the first premise, we can conclude $(V_1\phi_1 0)\theta_1$ is implied by A from the assumption. Then, θ_1 must be A-local or *global*. Thus, x and t_1 can not be B-local. Based on the assumption, $i\theta_2$ when simplified is not B-local. Thus, by replacing x(not B-local) by t_1(not B-local), $i\theta_2'$ when simplified is not B-local.
   (b) x is larger than any term any in t_1 and t_2. So, if $i\theta_2'$ is A-local, then $i\theta_2$ is A-local. Based on the assumption that the second premise is (A, B)-correct, we can conclude i is identical to $V_2\phi_2 0$. Thus, this property is true.
   (c) if the conclusion's interpolant part is identical to its constraint, it implies that the right premise's interpolant part is identical to its constraint because those two parts remain the same after applying the inference rule. Thus, based on the assumption that the second premise is (A, B)-correct, we can conclude i is identical to $V_2\phi_2 0$.

(d) Since x is larger than any term t_1 and t_2, and the first premise is implied by A, then t_1 and t_2 are A-local or global. So, θ_2' is A-local or global. Thus, the conclusion is geographically correct.

Therefore, the conclusion is (A, B)-correct.

□

The next rule we will define is the Combination rule, which applies when there is a Rewriting Rule but the Triple-Rewriting Rule defined above is not applicable. This Combination rule will involve applying the substitution before the inference and then reconstructing it after the inference. So we define the following two rules, which will be used in the Combination Rule. The first is **Back Substitution**:

$$\frac{c|i|\theta}{c\theta|i\theta|id}$$

In the conclusion, id is the identity substitution, since no variables exist in the constraint and interpolant part.

Lemma 5.7 *If the premise of Back Substitution is (A, B)-correct, then the conclusion is too.*

Proof.

(i) Based on the assumption, $A \models i\theta$. So, the first property is true.

(ii) Based on the assumption, $B \wedge i \models c$. Then, $B \wedge i\theta \models c\theta$. So, the second property $B \wedge i\theta \models c\theta$ is true.

(iii) Based on the assumption, we can easily derive that the conclusion is geographically correct. □

The second rule is **Reconstruct Substitution**.

$$\frac{x\phi t|i|id}{V\phi 0|V\phi^n x - t \wedge i|\{V \mapsto x - t\}}$$

Lemma 5.8 *If the premise of Reconstruct Substitution is (A, B)-correct then the conclusion is too.*

Proof. Since the idea of this inference rule is very similar to the second rule of Make-Triple, the proof is also similar. □

For a triple $c|i|\theta$, define $Apply(c|i|\theta)$ as $c\theta$. Given two triples t_1 and t_2, we want to define an inference rule whenever there is an inference between $Apply(t_1)$ and $Apply(t_2)$. Depending on the form of the interpolant part of the left premise, the Triple-Rewriting Rule will sometimes be applicable. In the Combination Rule, we define what to do when it is not applicable. The **Combination Rule** is the following inference

$$\frac{c_1|i_1|\theta_1 \quad c_2|i_2|\theta_2}{c|i_1\theta_1 \wedge i_2\theta_2|id}$$

where c is the conclusion of a Rewriting Rule between $c_1\theta_1$ and $c_2\theta_2$, and Reconstruct Substitution is immediately performed after this inference.

So we perform a Back Substitution followed by an inference on the constraints where the interpolants are conjoined, and then Reconstruct Substitution.

Lemma 5.9 *If the premises of the Combination Rule are (A, B) correct then the conclusion is too.*

Proof.

(i) Based on the assumption, $A \models i_1\theta_1$ and $A \models i_2\theta_2$. Thus, the first property $A \models i_1\theta_1 \wedge i_2\theta_2$ is true.

(ii) From the assumption, $B \wedge i_1 \models c_1$ and $B \wedge i_2 \models c_2$. Then, $B \wedge i_1 \wedge i_2 \models c_1 \wedge c_2$. Then, $(B \wedge i_1 \wedge i_2)\theta_1\theta_2 \models (c_1 \wedge c_2)\theta_1\theta_2$. Then, $B \wedge i_1\theta_1 \wedge i_2\theta_2 \models c_1\theta_1 \wedge c_2\theta_2$. Since c is the conclusion of $c_1\theta_1$ and $c_2\theta_2$, the second property $B \wedge i_1\theta_1 \wedge i_2\theta_2 \models c$ is true.

(iii) Next is to show the conclusion is geographically correct. Let θ_1 be $\{x - t_1\}$ and θ_2 be $\{x - t_2\}$.

(a) $i_1\theta_1$ and $i_2\theta_2$ when simplified are not B-local based on the assumption. Thus, $i_1\theta_1 \wedge i_2\theta_2$ when simplified is not B-local.

(b) If $i_1\theta_1 \wedge i_2\theta_2$ is A-local, then $i_1\theta_1$ or $i_2\theta_2$ is A-local. Suppose $i_1\theta_1$ is A-local. Based on the assumption, we can conclude i_1 and c_1 are identical. Since x is larger than any term in t_1, x must be A-local. So, $c_1\theta_1$ is from A. Similarly, we can conclude $c_2\theta_2$ is from A. But, if both premises are from A, the Triple-Rewriting Rule is applicable. So, we do not perform the Combination Rule. Therefore, $i_1\theta_1 \wedge i_2\theta_2$ can not be A-local.

(c) c and $i_1\theta_1 \wedge i_2\theta_2$ can not be identical. This property is obviously true.

(d) it is trivial that id does not contain A-local and B-local constants simultaneously.

Therefore, the conclusion is (A, B)-correct. □

Next, we show an example to construct an interpolant. We still use the example given in the previous part. Given two sets $A = \{x = y, x \geq 3z\}$, $B = \{y < z, z = 1\}$, the ordering of the constants is $z \prec y \prec x$.

Example 5.10 First, we apply Make-Triple to convert single constraints into triples.

$$\frac{x = y}{V_1 = 0 | V_1 = 0 | \{V_1 \mapsto x - y\}}$$

$$\frac{x \geq 3z}{V_2 \geq 0 | V_2 \geq 0 | \{V_2 \mapsto x - 3z\}}$$

$$\frac{y < z}{V_3 < 0 | V_3 \leq y - z | \{V_3 \mapsto y - z\}}$$

$$\frac{z = 1}{V_4 = 0 | V_4 = z - 1 \{V_4 \mapsto z - 1\}}$$

Then, we apply Triple-Rewriting Rule, Triple-Rewriting Rule, Simplify Rule, and Combination Rule in order.

$$\frac{V_1 = 0 | V_1 = 0 | \{V_1 \mapsto x - y\} \quad V_2 \geq 0 | V_2 \geq 0 | \{V_2 \mapsto x - 3z\}}{V_2 \geq 0 | V_2 \geq 0 | \{V_2 \mapsto y - 3z\}}$$

$$\frac{V_2 \geq 0 | V_2 \geq 0 | \{V_2 \mapsto y - 3z\} \quad V_3 < 0 | V_3 \leq y - z | \{V_3 \mapsto y - z\}}{V_3 < 0 | V_3 \leq y - z | \{V_3 \mapsto 2z\}}$$

$$\frac{V_3 < 0 | V_3 \leq y - z | \{V_3 \mapsto 2z\}}{2V_5 < 0 | 2V_5 \leq y - z | \{V_5 \mapsto z\}}$$

$$\frac{V_4 = 0 | V_4 = z - 1 \{V_4 \mapsto z - 1\} \quad V_5 < 0 | V_5 \leq \frac{1}{2}y - \frac{1}{2}z | \{V_5 \mapsto z\}}{\bot | z - 1 = z - 1 \wedge z \leq \frac{1}{2}y - \frac{1}{2}z | id}$$

Usually, we immediately perform Reconstruct Substitution after applying the Combination Rule. However, in this example since the constraint has become $\bot$, we will not perform Reconstruct Substitution. After simplifying the interpolant part, we derive $y \geq 3z$ which is an interpolant in this example.

Theorem 5.11 *If $\bot | i | \theta$ is derived using the inference rules defined in this section from sets A and B, then $i\theta$ is an (A, B)-interpolant.*

Proof. We have proved that each conclusion of our inference rules is (A, B)-correct. Thus, $\bot | i | \theta$ is (A, B)-correct.

(i) Since the triple is (A, B)-correct, $A \models i\theta$.

(ii) Since the triple is (A, B)-correct, $B \wedge i \models \bot$. So, $B \wedge i\theta \models \bot$

(iii) Since the triple is (A, B)-correct, $i\theta$ when simplified is not B-local and $i\theta$ is not A-local because either $\bot$ and i are not identical or i is $\bot$ and $i\theta$ is global. Thus, $i\theta$ contains only global constants.

Therefore, $i\theta$ is an (A, B)-interpolant.

□

We define a triple t to be *redundant in* S if $Apply(t)$ is redundant in $Apply(S)$, where $Apply(S) = \{Apply(t) \mid t \in S\}$. Then we define saturation of a set of triples just like saturation of a set of constraints. In other words, a set S of triples is saturated if $Apply(S)$ is *saturated*. Note that in S, we may have two different triples t_1 and t_2 where $Apply(t_1)$ is the same as $Apply(t_2)$. By completeness of constraints, it follows that for any set of triples, if S is saturated then $\bot \in Apply(S)$. The inference rules are also sound, because they simulate the sound inference rules on constraints.

6 Conclusion

We gave a method of constructing interpolants for linear constraints over the rationals. This is not in itself a new result. It has been done by McMillan. However, we believe our paper is novel in two ways. First is the actual inference system we use. It is based on rewriting techniques, which we believe will be more efficient than the method introduced by McMillan [4]. His approach requires inferences between any two constraints, whereas our approach orients the constraints so that the largest term is on the left hand side, and we only allow inferences which replace a constraint with a smaller one. For example, McMillan would apply an inference between constraints $0 \leq a$ and $0 \leq b$ to generate $0 \leq a + b$, whereas we would not do that.

Because our approach is based on rewriting, we can extend our method to handle function symbols without using a combination method. But, McMillan axiomatizes function symbols and gives a combination method to combine his function symbol inferences with the linear constraint inferences. We believe it is more efficient to deal with them all in the same framework, and not to axiomatize function symbols.

Aside from the relationship with McMillan's work, others[8,9] have examined combination methods for interpolants. In particular, [7] does not create the interpolant from the proof.

We plan on many directions for future work. We are currently extending our ideas, with the hope to give a complete method for linear constraints over integers with uninterpreted function symbols. We would also like to extend our framework to other theories. We are planning an implementation. Finally, we would like a better understanding of how our methods fits into invariant generation, predicate abstraction, and other uses of interpolants

References

[1] W. Craig. Three uses of the Herbrand-Gentzen theorem in relating model theory and proof theory. *J. Symbolic Logic*, 22(3), pages 269285. 1957.

[2] N. Dershowitz and D.A. Plaisted. Rewriting. In J.A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, Volume I, pages 537-608. Elsevier Science Publishers and MIT press, 2001.

[3] K.L. McMillan. Interpolation and SAT-based model checking. In *CAV'2003: Computer Aided Verification*, LNCS 2725, pages 1-13. Springer, 2003.

[4] K.L. McMillan. An interpolating theorem prover. In *TACAS'2004: Tools and Algorithms for the Construction and Analysis of Systems*, LNCS 2988, pages 16-30. Springer, 2004.

[5] K.L. McMillan. Applications of Craig interpolants in model checking. In *TACAS'2005: Tools and Algorithms for the Construction and Analysis of Systems*, LNCS 3440, pages 1-12. Springer, 2005.

[6] R. Nieuwenhuis and A. Rubio. Paramodulation-Based Theorem Proving. In J.A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, Volume I, pages 373-440. Elsevier Science Publishers and MIT press, 2001.

[7] A. Rybalchenko and V.S. Stokkermans. Constraint Solving for Interpolation. In *VMCAI'2007: Verification, Model Checking, and Abstract Interpretation*, LNCS 4349, pages 346-362. 2007.

[8] V.S. Stokkermans. Interpolation in local theory extensions. *Proceedings of IJCAR 2006*, Seattle, USA, LNAI 4130, pages 235-250. Springer, 2006.

[9] G. Yorsh and M. Musuvathi. A combination method for generating interpolants. In R. Nieuwenhuis, editor, *20th International Conference on Automated Deduction (CADE-20)*, LNAI 3632, pages 353-368. Springer, 2005.