The Relative Chain Framework for Modular Termination

Tom.Divers@bristol.ac.uk, Eddie.Jones@bristol.ac.uk

1 Introduction

We present a new framework of **relative chains** for proving the termination of the union of two terminating rewrite systems. Our approach extends the dependency pair framework, introduced by Arts and Giesl in [1]. We demonstrate how our framework can be used to prove the termination of functional programs augmented with equational hypotheses, and **...**

2 Preliminaries

In this section, we discuss some mathematical preliminaries concerning term rewriting and reduction relations. The foundations of this field are explored in depth in [2].

2.1 Term Rewrite Systems

We consider the setting of term rewriting systems over a **signature** $T(\Sigma, V)$. A signature consists of a finite set of **function symbols** Σ and an infinite set of **variables** V. Σ is presumed to be partitioned into a set of **defined functions** Σ_{def} and **constructors** Σ_{con}. Each $f \in \Sigma$ has an associated natural number called its **arity**, and $\Sigma^{(i)}$ denotes all the function symbols in Σ with arity i.

$T(\Sigma, V)$ is defined inductively as follows:

1. $\Sigma^{(0)} \subseteq T(\Sigma, V)$ and $V \subseteq T(\Sigma, V)$.
2. If $f \in \Sigma^{(n)}$, and $M_i \in T(\Sigma, V)$ for each $i \in [n]$, then $f(M_1, \cdots, M_n) \in T(\Sigma, V)$.

The recursive structure of $T(\Sigma, V)$ prompts us to think of terms in a TRS as **syntax trees**. We define $Pos(M) \subseteq \mathbb{N}^*$ for each $M \in T(\Sigma, V)$ to be the set of positions in M's syntax tree. The root of the tree has position ϵ, and the ith child of a node at position p has position pi. $M|_p$ is the subterm of M rooted at position p, and $M[N]_p$ denotes M with its term at position p replaced by $N \in T(\Sigma, V)$. Additionally, we define $Var(M) \subseteq V$ to be the set of variables appearing in a term.

A **(one-hole) context** $C[\cdot] : T(\Sigma, V) \to T(\Sigma, V)$ is a term with a hole $\Box$ at some position. $C[M]$ denotes the context $C[\cdot]$ with its hole replaced by the term M. A **substitution** $\theta : X \to T(\Sigma, V)$ is a function mapping variables to terms, for which the set $dom(\theta) := \{x \mid \theta(x) \neq x\}$ is finite. Therefore, a substitution can be defined in terms of its $dom(\theta)$-images, in which case we write $\theta = [x_1 \mapsto M_1, x_2 \mapsto M_2, \cdots]$ for variables x_i and terms M_i. $M\theta$ denotes the term M with each variable replaced by its θ-image, and we call $M\theta$ an **instance** of M.

A reduction system $(X, \to)$ consists of a set X equipped with a binary relation $\to \subseteq X \times X$. A reduction is **terminating** iff there exist no infinite sequences $x_1 x_2 \cdots \in X^\omega$ with $x_1 \to x_2 \to \cdots$. $x \in X$ is a **redux** of $\to$ iff $\exists y \in X : x \to y$. If x is not a redux, we say that it is in **$\to$-normal form**. A reduction system over $T(\Sigma, V)$ is a **term rewrite syste (TRS)** iff it is closed under contexts and substitutions (i.e. if $M \to N$, then $C[M\theta] \to C[N\theta]$ for all contexts $C[\cdot]$ and substitutions θ).

Here, we will consider TRSs defined by a set of equations $R \subseteq T(\Sigma, V) \times T(\Sigma, V)$. Note that our equations are presumed to be **oriented**, meaning that $M \approx N \in R$ does not imply that $N \approx M \in R$. We also assume that $Var(M) \supseteq Var(N)$ for each $M \approx N \in R$. We define $\to_R$ such that, for each $M \approx N \in R$, and for all contexts $C[\cdot]$ and substitutions θ, $C[M\theta] \to_R C[N\theta]$. We say that an equation $M \approx N$ is **stable** iff M is headed by a defined function symbol. A TRS is stable iff all of its equations are stable.

2.2 The Dependency Pair Framework

The dependency pair framework [1] allows us to analyse the interdependencies between functions more precisely than by simply analysing the rules of a TRS. A dependency pair of a TRS R is written $\langle M, N \rangle_R \in T(\Sigma \cup \Sigma^\sharp, V)^2$. We define $\Sigma^\sharp$ such that, for each $f \in \Sigma$, we have $F \in \Sigma^\sharp$. These pairs are constructed as follows:

Definition 2.1. *Consider some TRS R, and assume that $f(s_1, \cdots, s_n) \approx C[g(s_1, \cdots, s_n)] \in R$ for some context $C[\cdot]$ and function symbols $f, g \in \Sigma_{def}$. This induces the **dependency pair** $\langle F(s_1, \cdots, s_n), G(s_1, \cdots, s_n) \rangle_R$, where $F, G \in \Sigma^\sharp$. $DP(R)$ is defined to be the set of all dependency pairs induced by R.*

The fact that each dependency pair is induced by exactly one equational constraint leads trivially to the following result:

Lemma 2.2. $DP(R_1 \cup R_2) = DP(R_1) \cup DP(R_2)$

In order to reason about the termination of a TRS through the analysis of its dependency pairs, we need to introduce the following definition, which allows us to connect reduction sequences of our TRSs to sequences of dependency pairs. Theorem 2.4 then asserts that termination can be equivalently characterised by the nonexistence of infinite chains.

Definition 2.3. *Consider some fixed TRS R. An **R-chain** is a sequence of dependency pairs $\langle s_1, t_1\rangle_R \langle s_2, t_2\rangle_R \cdots$, along with a sequence of substitutions $\theta_1\theta_2\cdots$, such that $t_i\theta_i \to_R^* s_{i+1}\theta_{i+1}$ for all i.*

Theorem 2.4 (Arts and Giesl [1])**.** *A TRS is terminating iff it does not induce an infinite chain.*

3 Relative Chains

In order to analyse the termination of the union of two TRSs, we consider the union of their dependency pairs. The following definition generalises the notion of a chain, separating rewrite system used in the intermediate reduction steps from the rewrite system that induces the dependency pairs in the chain.

Definition 3.1. *Consider two TRSs R and S. A sequence of R's dependency pairs $\langle s_1, t_1\rangle_R \langle s_2, t_2\rangle_R \cdots$ forms an **R/S-chain** iff there exists some sequence of substitutions $\theta_1\theta_2\cdots$ satisfying $t_i\theta_i \to_S^* s_{i+1}\theta_{i+1}$ for each i.*

We also define what it means for a chain over the union of two rewrite systems to alternate between applying rules in each of the systems:

Definition 3.2. *Consider some TRS $S := R_1 \cup R_2$, which is the union of two other TRSs. An S-chain is **spanning** iff it is of the form $\cdots p_1 p_2 \cdots$ where $p_1 \in DP(R_1)$ and $p_2 \in DP(R_2)$, or vice versa. In this case, we say that this chain spans **from** R_1 **to** R_2.*

The following theorem characterises modular termination in terms of sub-relative termination and the nonexistence of two-way spanning chains:

Theorem 3.3. *Consider two TRSs R_1 and R_2. $S := R_1 \cup R_2$ is terminating if both of the following hold:*

1. *There are no infinite R_1/S- or R_2/S-chains.*
2. *Every spanning S-chain only spans from R_1 to R_2 (or vice versa).*

Proof. We proceed by contrapositive. Assume that S is nonterminating; hence, by Theorem 2.4, S induces an infinite chain. We now demonstrate that any such chain violates one of our conditions. Observe that, since $DP(S)$ is finite, an infinite chain must repeat some finite cycle $s \in DP(S)^*$ infinitely many times; additionally, assume that s is the longest such cycle. Also note that, via Lemma 2.2, $DP(S) = DP(R_1) \cup DP(R_2)$. Hence, this cycle must occur either entirely within $DP(R_1)$ or $DP(R_2)$, or must alternate between $DP(R_1)$ and $DP(R_2)$. More formally, we have two cases:

1. $s \in DP(R_1)^*$ or $s \in DP(R_2)^*$. Hence, R_1 or R_2 induces an infinite chain, which violates condition (1).
2. $s = \cdots x_1 x_2 \cdots y_2 y_1 \cdots$, where $x_i, y_i \in DP(R_i), i \in \{1, 2\}$ (i.e. s spans from R_1 to R_2 and from R_2 to R_1), which violates condition (2).

□

4 Flat Termination

Definition 4.1. *Consider some TRS P over $T(\Sigma, V)$, which induces the dependency pairs $\langle\cdot,\cdot\rangle_P$ over $T(\Sigma \cup \Sigma^\sharp, V)$. Also assume that we have access to an infinite set of unused **flat variables** $V^\flat \subseteq V$.*

*We define a set of **flat dependency pairs** $\langle\cdot,\cdot\rangle_P^\flat$, where each rule is obtained by **flattening** a dependency pair. For a dependency pair $\langle s, G(t_1, \cdots, t_n)\rangle_P$, its flattening is written $\langle s, G(t_1^\flat, \cdots, t_n^\flat)\rangle_P^\flat$. A term t's flattening $t^\flat$ is computed recursively as follows (where $x^\flat \in V^\flat$ is an arbitrary unused flat variable):*

- *If $t \in \Sigma_{con}$, then $t^\flat := t$*
- *If $t \in V$ or $t = f(b_1, \cdots, b_k)$ where $f \in \Sigma_{def}$, then $t^\flat := x^\flat$.*
- *If $t = f(b_1, \cdots, b_k)$ where $f \in \Sigma_{con}$, then $t^\flat := f(b_1^\flat, \cdots, b_k^\flat)$.*

*$P^\flat$- and $P^\flat/Q$-chains are defined as before, and a TRS P is **flat-terminating** iff it induces no infinite $P^\flat$-chains.*

Lemma 4.2. *A TRS P is flat-terminating iff it induces no infinite $P^\flat/=$-chains.*

Proof. □

Theorem 4.3. *Consider a flat-terminating TRS P. There are no infinite $P^\flat/Q$-chains for any TRS Q that is P-normal.*

Proof. □

5 Existing Results in the Relative Chain Framework

We begin with a couple of definitions:

Definition 5.1. *An equation $M \approx N$ is **collapsing** iff N is a variable.*

Definition 5.2. *A TRS R over $T(\Sigma, V)$ is $\mathcal{C}_\epsilon$-**terminating** iff $R \cup \{\chi(x,y) \approx x, \chi(x,y) \approx y\}$ is terminating, where $\chi \in \Sigma^{(2)}_{con}$ does not appear in R.*

We attempt to demonstrate the versatility and applicability of our framework by applying it to prove the following existing result:

Theorem 5.3 (Ohlebusch [3])**.** *Consider two disjoint and terminating TRSs R_1 and R_2. If R_1 is $\mathcal{C}_\epsilon$-terminating or R_2 is non-collapsing, then $S := R_1 \cup R_2$ is terminating.*

Proof. We have two cases:

1. R_1 is $\mathcal{C}_\epsilon$-terminating.
2. R_2 is non-collapsing.

We prove termination in each case separately.

□

References

[1] Thomas Arts and Jürgen Giesl. "Termination of term rewriting using dependency pairs". In: *Theoretical Computer Science* 236.1 (2000), pp. 133–178. DOI: 10.1016/S0304-3975(99)00207-8.

[2] Franz Baader and Tobias Nipkow. *Term Rewriting and All That*. Cambridge University Press, 1998. DOI: 10.1017/CBO9781139172752.

[3] Enno Ohlebusch. "On the modularity of termination of term rewriting systems". In: *Theoretical Computer Science* 136.2 (1994), pp. 333–360. DOI: 10.1016/0304-3975(94)00039-L.