On Forward Closure and the Finite Variant Property*

Christopher Bouchard[1], Kimberly A. Gero[1],
Christopher Lynch[2], and Paliath Narendran[1]

[1] University at Albany—SUNY, Albany, NY, USA
{cbou,kgero001,dran}@cs.albany.edu
[2] Clarkson University, Potsdam, NY, USA
clynch@clarkson.edu

Abstract. Equational unification is an important research area with many applications, such as cryptographic protocol analysis. Unification modulo a convergent term rewrite system is undecidable, even with just a single rule. To identify decidable (and tractable) cases, two paradigms have been developed — Basic Syntactic Mutation [14] and the Finite Variant Property [6]. Inspired by the Basic Syntactic Mutation approach, we investigate the notion of forward closure along with suitable redundancy constraints. We show that a convergent term rewriting system R has a finite forward closure if and only if R has the finite variant property. We also show the undecidability of the finiteness of forward closure, therefore determining if a system has the finite variant property is undecidable.

Keywords: Equational unification, Finite variant property, Forward closure, Term rewriting, Undecidability.

1 Introduction

Equational unification is an important research area which has applications in cryptographic protocol analysis, automated theorem proving, and automated reasoning. However, unification modulo a convergent term rewrite system is undecidable in general, even if the system has just a single rule [1]. Consequently, there is interest in identifying decidable instances of equational unification. Two important syntactic paradigms have been developed to identify such instances. One paradigm was developed in "Basic Syntactic Mutation", by Christopher Lynch and Barbara Morawska [14]. They give syntactic criteria on equational axioms E which guarantee that the corresponding E-unification problem is in **NP**. If the system satisfies some additional criteria, they provide a polynomial-time decision algorithm for that E-unification problem. The second paradigm was developed in "The finite variant property: How to get rid of some algebraic

* C. Bouchard, K. Gero, and P. Narendran were supported in part by NSF grant CNS 09-05286. C. Lynch was supported in part by NSF grant CNS 09-05378.

P. Fontaine, C. Ringeissen, and R.A. Schmidt (Eds.): FroCos 2013, LNAI 8152, pp. 327–342, 2013.

properties" by Hubert Comon-Lundh and Stéphanie Delaune [6]. Here it was shown that E-unification is decidable if E has the finite variant property, and Escobar, Meseguer, and Sasse showed how narrowing can be used to implement an E-unification decision algorithm for such an E [9].

In studying the BSM algorithm in the context of convergent rewrite systems [5], we found that the notion of saturation by paramodulation is equivalent to that of forward closure if the system is convergent and suitable redundancy constraints are added. Hermann considers the idea of forward closure chains in "Chain Properties of Rule Closures" [10], and he proved that the finiteness of forward closure is undecidable for general rewrite systems—in particular, the system he considers has an undecidable termination problem. Hermann did not, however, consider any sort of redundancy.

In this paper, we extend the notion forward closure[1] to allow redundancy constraints and show that a convergent term rewriting system R has a finite forward closure if and only if R has the finite variant property. In showing this equivalence we define the IR-boundedness property which characterizes the finite variant property. Additionally, we show the undecidability of the finiteness of forward closure for *convergent* rewrite systems, and therefore that determining if a system has the finite variant property for such systems is undecidable. Finally, we show that the finiteness of forward closure is a modular property, i.e., if two disjoint rewrite systems have a finite forward closure, their union also has a finite forward closure.

In the interest of space, several proofs and examples have been omitted or shortened in this version. They are given in full in a tech report [4].

2 Notation and Preliminaries

We consider rewrite systems over ranked signatures, usually denoted Σ, and a possibly infinite set of variables, usually denoted $\mathcal{X}$. We assume the reader is familiar with the usual notions and concepts in term rewriting systems [2] and equational unification [3]. The set of all terms over Σ and $\mathcal{X}$ is denoted as $T(\Sigma, \mathcal{X})$. Given a term t, we denote by $\mathcal{P}os(t)$ the set of all positions in t, and by $\mathcal{FP}os(t)$ the set of all non-variable positions in t. An *equation*, e.g. in [2] is an ordered pair of terms $(s,\, t)$, usually written as $s \approx t$. Here s is the *left-hand side* and t is the *right-hand side* of the equation [2]. A *rewrite rule* is an equation $s \approx t$ where $\mathcal{V}ar(s) \supseteq \mathcal{V}ar(t)$, usually written as $s \to t$. A *term rewriting system* is a set of rewrite rules.

Our focus in this paper is on unifiability modulo theories that have convergent term rewriting systems. Let R be a convergent term rewriting system. We assume that there is a well-founded reduction ordering $\succ$ on terms such that $\to_R^+ \subseteq \succ$. Let $\prec$ be the inverse of $\succ$, i.e., $s \prec t$ if and only if $t \succ s$. We further assume that the ordering is total on ground terms. We extend this order to equations as $(s \approx t) \succ (u \approx v)$ if and only if $\{s, t\} \succ_{mul} \{u, v\}$, where $\succ_{mul}$ is the multiset

[1] From this point on, we will use "forward closure" to mean "forward closure *with redundancy constraints*".

order induced by $\succ$. A term t is an *innermost redex* of a rewrite system R if and only if all proper subterms of t are irreducible and t is an instance of the left-hand side of a rule in R.

The following proposition holds since $\rightarrow_R \subseteq \succ$ and since $\succ$ is transitive.

Proposition 1. *Let R be a convergent rewrite system and let t, l, and r be terms such that $t \succ l$ and $t \succ r$. If $l \downarrow_R r$, then every term that appears in the rewrite proof ("valley proof") is below t in the reduction ordering $\succ$.*

3 Strict Redundancy

Given a set of equations E, the set of ground instances of equations in E is denoted by $Gr(E)$. An instance is ground if its terms do not contain any variables. A ground equation e is *strictly redundant in* E if and only if it is a consequence of equations in $Gr(E)$ which are smaller than e modulo the ordering we use to show termination [14]. An equation e is *strictly redundant in* E if and only if every ground instance e' of e is strictly redundant in E. In our setting, with convergent rewriting systems R and reduction orderings $\succ$, this can be formulated as follows. For a ground equation $s \approx t$ we define the following (possibly infinite) ground term rewriting system:

$$\mathcal{G}_R^{\prec(s\approx t)} := \{l \rightarrow r \mid (l \rightarrow r) \in Gr(R) \text{ and } (l \rightarrow r) \prec (s \approx t)\}$$

Now a ground equation $s \approx t$ is strictly redundant in R if and only if

$$\mathcal{G}_R^{\prec(s\approx t)} \;\vdash\; s \approx t$$

Since our focus in this paper is on convergent rewrite systems, we first give a condition on R such that $\mathcal{G}_R^{\prec(s\approx t)}$ is convergent.

Lemma 1. *Let R be a convergent rewrite system, and let s and t be ground terms such that $s \succ t$. Then $\mathcal{G}_R^{\prec(s\approx t)}$ is convergent.*

Now we explore conditions on equations that force those equations to be redundant in a rewrite system. The following lemma follows almost directly from the definition of $\mathcal{G}_R^{\prec(s\approx t)}$.

Lemma 2. *Let R be a convergent rewrite system. Then an equation $s_1 \approx s_2$ is strictly redundant in R if and only if for every ground instance $\delta(s_1) \approx \delta(s_2)$ of $s_1 \approx s_2$, $\delta(s_1)$ and $\delta(s_2)$ are joinable modulo $\mathcal{G}_R^{\prec(\delta(s_1)\approx\delta(s_2))}$.*

Lemma 3. *Suppose R is a convergent rewrite system such that the rule $l \rightarrow r$ is strictly redundant in R. Then the rule $\theta(l) \rightarrow \theta(r)$ is strictly redundant in R for any substitution θ.*

Lemma 4. *Let R be a convergent rewrite system, and let l and r be terms joinable modulo R such that $l \succ r$ and a proper subterm of l is reducible. Then $l \approx r$ is strictly redundant in R.*

Proof. Suppose $l \approx r$ is not strictly redundant in R. Then, by Lemma 2, there is a ground instance $\delta(l) \approx \delta(r)$ such that $\delta(l)$ and $\delta(r)$ are not joinable in $\mathcal{G} = \mathcal{G}_R^{\prec(\delta(l)\approx\delta(r))}$. Since a proper subterm of l is reducible, there is a rule $l' \rightarrow r'$ in R and a position $p \neq \epsilon$ in $\mathcal{P}os(l)$ such that $l|_p = \sigma(l')$ and $l \rightarrow_R l[\sigma(r')]_p$. Therefore $\delta(l)|_p = \delta(\sigma(l'))$ and $\delta(l) \rightarrow_R \delta(l[\sigma(r')]_p)$. Since $\rightarrow_R \subseteq \succ$, we have that $\delta(l) \succ \delta(l[\sigma(r')]_p)$, and since reduction orders are closed under substitutions, $\delta(l) \succ \delta(r)$. Thus, by Proposition 1, there are rewrite sequences

$$\delta(r) \rightarrow_R^* t \leftarrow_R^* \delta(l[\sigma(r')]_p)$$

such that each term in the rewrite sequences is below $\delta(l)$ in the ordering $\prec$. Therefore, since each term is ground, $\delta(r)$ and $\delta(l[\sigma(r')]_p)$ are joinable in $\mathcal{G}$.

Since $\delta(l) = \delta(l[\sigma(l')]_p) = \delta(l)[\delta(\sigma(l'))]_p$, and since the ordering $\prec$ has the subterm property on ground terms, $\delta(\sigma(l)) \prec \delta(l)$. Thus $\delta(l) \rightarrow_{\mathcal{G}} \delta(l[\sigma(r')]_p)$. So $\delta(l)$ and $\delta(r)$ are joinable in $\mathcal{G}$, which is a contradiction. Therefore $s \approx r$ is strictly redundant in R. □

Lemma 5. *Let R be a convergent rewrite system and $l \approx r$ be an equation such that l is an innermost redex and $l \rightarrow_R^+ r$. Then $l \approx r$ is strictly redundant in R if there is a term r' such that $l \rightarrow_R r'$ and $r' \prec r$.*

Proof. If $l \rightarrow_R r'$ and l is an innermost redex, then $l \rightarrow r'$ is an instance of a rule in R and $(l \approx r') \prec (l \approx r)$. Since R is confluent, $r' \downarrow_R r$ and, by Proposition 1, every term that appears in the rewrite proof is below l in the ordering. Thus every ground instance of $l \rightarrow r$ can be proven using only smaller instances of rules in R, and therefore $l \approx r$ is strictly redundant in R. □

Unfortunately, the converse cannot be proved unless additional assumptions are made about the ordering. However, for ground equations we can prove both directions:

Lemma 6. *Let R be a convergent rewrite system and $l \approx r$ be a* ground *equation such that l is an innermost redex and $l \rightarrow_R^+ r$. Then $l \approx r$ is strictly redundant in R if and only if there is a ground term r' such that $l \rightarrow_R r'$ and $r' \prec r$.*

Proof. The "if" part follows from Lemma 5.

Suppose now that there is no term $r' \prec r$ such that $l \rightarrow_R r'$, but $l \approx r$ is strictly redundant in R. Then by Lemma 2, l and r must be joinable modulo $\mathcal{G}_R^{\prec(l\approx r)}$. Thus there must be a rule $l \rightarrow r''$ in $\mathcal{G}_R^{\prec(l\approx r)}$, and so $(l \rightarrow r'') \prec (l \approx r)$. We then have that $r'' \prec r$. This is a contradiction, so $l \approx r$ is not strictly redundant in R. □

This leads us to a very useful lemma. In practice many of the equations we look at will be rewrite rules whose right-hand side is in normal form. This gives us a simple syntactic check for the redundancy of such rules.

Lemma 7. *Let R be a convergent rewrite system, and let $l \approx r$ be an equation such that l is reducible and r is the normal form of l. Then $l \approx r$ is strictly redundant in R if and only if a proper subterm of l is reducible.*

4 A (Slightly) Stronger Notion of Redundancy

A rule $\rho_1 = l_1 \to r_1$ is said to be an instance of a rule $\rho_2 = l_2 \to r_2$ if and only if there is a substitution σ such that $\sigma(l_2) = l_1$ and $\sigma(r_2) = r_1$. We write this as $\rho_2 \sqsupseteq \rho_1$ or as $\rho_2 \sqsupseteq_\sigma \rho_1$ if the substitution σ is of significance. For instance, the rule $f(x, x) \to x$ is an instance of the rule $f(x, y) \to x$.

A rule ρ is *redundant*[2] in R if and only if it is either strictly redundant in R (i.e., every ground instance of ρ is strictly redundant in R) or there is a rule ρ' in R such that $\rho' \sqsupseteq \rho$.

We can extend Lemma 3 from the previous section to redundancy as follows.

Lemma 8. *Let R be a convergent rewrite system such that the rule $l \to r$ is redundant in R. Then the rule $\theta(l) \to \theta(r)$ is redundant in R for any substitution θ.*

5 Forward Closure

Following Hermann [10], the *forward-closure* of a term rewrite system R is defined in terms of the following operation on rules in R. Let $\rho_1 = l_1 \to r_1$ and $\rho_2 = l_2 \to r_2$ be two rules in R, and let $p \in \mathcal{FP}os(r_1)$. Then

$$\rho_1 \rightsquigarrow_p \rho_2 \;:=\; \sigma(l_1 \to r_1[r_2]_p)$$

where $\sigma = mgu(r_1|_p =^? l_2)$. We call this the *forward overlap* of ρ_1 and ρ_2 at p.

Proposition 2. *Let ρ_1, ρ_2, and ρ_3 be rules such that $\rho_3 = \rho_1 \rightsquigarrow_p \rho_2$ for some position p. If $t \to_{\rho_3} t'$ then $\exists t'' \colon t \to_{\rho_1} t''$ and $t'' \to_{\rho_2} t'$.*

Given rewrite systems R_1, R_2, and R_3 we define $\mathcal{FOV}(R_1, R_2)$ (the set of forward overlaps) and $\mathcal{N}(R_1, R_2, R_3)$ (the set of non-redundant rules) as

$$\mathcal{FOV}(R_1, R_2) := \{\rho_1 \rightsquigarrow_p \rho_2 \mid \rho_1 = (l_1 \to r_1) \in R_1,\ \rho_2 \in R_2, \text{ and } p \in \mathcal{FP}os(r_1)\}$$
$$\mathcal{N}(R_1, R_2, R_3) := \{\rho \mid \rho \in \mathcal{FOV}(R_1, R_2) \text{ and } \rho \text{ is not redundant in } R_3\}$$

We now simultaneously define $NR_k(R)$ (new rules step) and $FC_k(R)$ (forward closure step) for all $k \geq 0$.

$$\begin{aligned} NR_0(R) &:= R & NR_{k+1}(R) &:= \mathcal{N}(NR_k(R), R, FC_k(R)) \\ FC_0(R) &:= R & FC_{k+1}(R) &:= FC_k(R) \cup NR_{k+1}(R) \end{aligned}$$

Finally, we define the forward closure of R.

$$FC(R) := \bigcup_{i=1}^{\infty} FC_i(R)$$

Note that $FC_k(R) \subseteq FC_{k+1}(R)$ for all $k \geq 0$. A set of rewrite rules R is *forward-closed* if and only if $FC(R) = R$.

[2] This is referred to as *non-strictly redundant* in [15].

Example 1. The following rewrite system has a finite forward closure:

$$R_{\text{ex}} = \{f(s(x)) \to f(x),\ s(s(s(x))) \to x\}$$

There is an overlap of the first rule with itself, and we see that the rewrite system has one forward overlap,

$$\mathcal{FOV}(NR_0(R_{\text{ex}}),\, R_{\text{ex}}) = \{f(s(s(x))) \to f(x)\}$$

This rule is not redundant in R_{ex}, as the ground instance $f(s(s(a))) \approx f(a)$ cannot be proven by $\mathcal{G}_{R_{\text{ex}}}^{\prec(f(s(s(a)))\approx f(a))}$, i.e. smaller rules in $Gr(R_{\text{ex}})$. Thus we see that

$$NR_1(R_{\text{ex}}) = \{f(s(s(x))) \to f(x)\}$$
$$FC_1(R_{\text{ex}}) = \{f(s(s(x))) \to f(x),\ f(s(x)) \to f(x),\ s(s(s(x))) \to x\}$$

To compute the next set of forward overlaps, we can only overlap the new rule with the first rule of R_{ex}. So there is one new forward overlap,

$$\mathcal{FOV}(NR_1(R_{\text{ex}}),\, R_{\text{ex}}) = \{f(s(s(s(x)))) \to f(x)\}$$

However, this rule is redundant by Lemma 7, since the subterm $s(s(s(x)))$ at position 1 of the left-hand side is reducible. Thus $NR_2(R_{\text{ex}}) = \emptyset$, and the rewrite system has a finite forward closure $FC(R_{\text{ex}}) = FC_1(R_{\text{ex}})$. □

Now we will give constraints that must be satisfied to have a finite forward closure.

Lemma 9. *Given a convergent rewrite system R, $FC(R)$ is finite if and only if there is a $k > 0$ such that $NR_k(R) = \emptyset$.*

Corollary 1. *Given a convergent rewrite system R, $FC(R)$ is finite if and only if there is a $k > 0$ such that $FC(R) = FC_k(R)$.*

Now we will discuss the case where a term t is an innermost redex.

Lemma 10. *Let R be a convergent rewrite system, and let t and t' be terms where t is an innermost redex. If $t \to_{FC_{k'}(R)} t'$ then $t \to_R^k t'$ for some $k \leq k'+1$.*

Proof. Suppose $k' = 0$. Then $FC_{k'}(R) = R$, and thus $t \to_R t'$.

Otherwise, assume that if $t \to_{FC_{k'-1}(R)} t'$ then $t \to_R^k t'$ for some $k \leq k'$. If $t \to_{FC_{k'}(R)} t'$ then either $t \to_{FC_{k'-1}(R)} t'$ or $t \to_{NR_{k'}(R)} t'$. In the first case we are done. In the second case, $t \to t'$ is in $NR_{k'}(R) = \mathcal{N}(NR_{k'-1}(R), R, FC_{k'-1}(R))$. Therefore $(t \to t') = \rho_1 \rightsquigarrow_p \rho_2$, for ρ_1 in $NR_{k'-1}(R)$, ρ_2 in R, and position p. Since $NR_{k'-1}(R) \subseteq FC_{k'-1}(R)$, $t \to_{FC_{k'-1}(R)} t'' \to_R t'$ for some t''. By our assumption, $t \to_R^k t''$ for some $k \leq k'$, so $t \to_R^{k+1} t'$. □

In the next lemma we show that when our initial rewrite system R is convergent then at every step in our forward closure procedure the rewrite system returned is convergent.

Lemma 11. *Let R be a convergent rewrite system. Then for all $k \geq 0$, $FC_k(R)$ is convergent.*

Throughout the remainder of the section we will show that our forward closure procedure will get an innermost redex "closer and closer" to its normal form. The section culminates in a theorem that will be used to show one of the main results in this paper.

Lemma 12. *Let R be a convergent rewrite system, and let t and t' be terms where t is a* ground *innermost redex and $t \rightarrow_{FC_k(R)} t'$ for some $k \geq 0$. If t' is not in normal form then there exists a term $t'' \prec t'$ such that $t \rightarrow_{FC_{k+1}(R)} t''$.*

Proof (Sketch). If t' is not in normal form, then there is some rule in $FC_k(R)$ that rewrites t to t'. This rule will be overlapped with a rule from R in the next step of forward closure, resulting in a new rule to a lower term. □

Lemma 13. *Suppose R is a convergent rewrite system and t an innermost redex with normal form $\hat{t}$ where $t \rightarrow_R^{k'} \hat{t}$. Then there is a k such that $t \rightarrow_{FC_k(R)} \hat{t}$.*

Proof. Let θ be a substitution that maps each variable x in t to a distinct free constant c_x. Let $s = \theta(t)$ and $\hat{s} = \theta(\hat{t})$. Note that $\theta(\hat{t})$ is still irreducible, so $\hat{s}$ is the normal form of s. Also note that, by Lemma 11, since R is convergent so is $FC_k(R)$ for any $k \geq 0$.

Suppose there is no k such that $s \rightarrow_{FC_k(R)} \hat{s}$. Then, by Lemma 12, if $s \rightarrow_{FC_k(R)} s_k$ for some k and some ground term s_k, then there is a ground term $s_{k+1} \prec s_k$ such that $s \rightarrow_{FC_{k+1}(R)} s_{k+1}$. Thus there is an infinitely descending chain

$$s \succ \cdots \succ s_k \succ s_{k+1} \succ s_{k+2} \succ \cdots$$

and therefore the ordering $\succ$ is not well-founded. This is a contradiction, so there must be a k such that $s \rightarrow_{FC_k(R)} \hat{s}$. Since $s = \theta(t)$ is an innermost redex, this rewrite occurs at the root. Thus there is a rule $\rho = (l \rightarrow r)$ in $FC_k(R)$ such that $\rho \sqsupseteq_\sigma (\theta(t) \rightarrow \theta(\hat{t}))$.

Suppose now that t does not rewrite to its normal form in one step modulo $FC_k(R)$. Then $\rho \not\sqsupseteq (t \rightarrow \hat{t})$. If $\theta \sqsupseteq_\tau \sigma$, then $\rho \sqsupseteq_\tau (\theta(t) \rightarrow \theta(\hat{t}))$ since $\theta \circ \theta = \theta$ (i.e., θ is idempotent). But then $\rho \sqsupseteq_\sigma (t \rightarrow \hat{t})$. So $\theta \not\sqsupseteq \sigma$. This means there is a position p in l such that $l|_p = c_x$ for some x. This is a contradiction since each c_x is free. Thus $t \rightarrow_{FC_k(R)} \hat{t}$. □

Corollary 2. *If R is a convergent rewrite system and t an innermost redex with normal form $\hat{t}$, then $t \rightarrow_{FC(R)} \hat{t}$.*

Theorem 1. *A convergent rewrite system R is forward-closed if and only if every innermost redex can be reduced to its R-normal form in one step.*

Proof. If $R = FC(R)$ then, by Corollary 2, for any innermost redex t with normal form $\hat{t}$, $t \rightarrow_R \hat{t}$. Thus we have proven the "only if" part.

To prove the "if" part, assume that every innermost redex can be reduced to its normal form in one step, but R is not forward-closed. Thus there is a rule

$l \rightarrow r$ in $FC(R)$ that is not in R. If l is not an innermost redex in R then, by Lemma 4, $l \rightarrow r$ is redundant in R. So l must be an innermost redex in R and can be reduced to its normal form $\hat{l}$ in one step. Since $(l \rightarrow \hat{l}) \prec (l \rightarrow r)$, and since R is confluent, l and r are joinable using only smaller instances of rules in R and thus $l \rightarrow r$ is redundant in R. This is a contradiction, so R must be forward-closed. □

6 Equivalence of Finiteness of Forward Closure and the Finite Variant Property

In this section we show that a system has a finite forward closure (with redundancy) if and only if it has the *finite variant property*, as defined by Comon-Lundh and Delaune [6]. We will adopt the notation used in [7].

Definition 1. *Let R be a convergent rewrite system. A term-substitution pair (t, θ) is an R-*variant *of a term s if and only if θ is R-normalized and $\theta(s) \rightarrow^!_R t$. An R-variant (t, θ) of a term s is said to be more general than another R-variant (t', θ') of the same term s, denoted as $(t, \theta) \sqsupseteq (t', \theta')$, if and only if there is a substitution ρ such that $t' = \rho(t)$ and $\theta' = \rho \circ \theta$. A complete set of R-variants of a term s, denoted as $[\![s]\!]^\star$, is a set of R-variants of s, such that for every R-variant (s', γ) of s there is a variant $(t, \theta) \in [\![s]\!]^\star$ such that $(t, \theta) \sqsupseteq (s', \gamma)$. A convergent term rewriting system R has the* finite variant property *if and only if every term s has a* finite *complete set of R-variants.*

Comon-Lundh and Delaune showed that the finite variant property is equivalent to the *boundedness* property.

Definition 2. *A rewrite system R has the boundedness property (or is bounded) if, for every term t, there exists an integer n such that for every normalized substitution σ, the normal form of $\sigma(t)$ is reachable by a derivation whose length can be bounded by n (thus independently of σ):*

$$\forall t \; \exists n \; \forall \sigma \colon \; (\sigma\!\downarrow)(t) \xrightarrow{\le n}_R \sigma(t)\!\downarrow$$

We first introduce a different notion of boundedness for a term rewriting system and prove that this new notion is equivalent to the standard notion.

Definition 3. *A rewrite relation $\rightarrow_R$ (alternatively, a term rewriting system R) is* IR-bounded *if and only if there is a "global" bound n such that every innermost redex can be reduced to its normal form in n steps or less:*

$$\exists n \; \forall t \colon \; \Big[t \text{ is an innermost redex} \Rightarrow t \xrightarrow{\le n}_R t\!\downarrow \Big]$$

Lemma 14. *Suppose a convergent rewrite system R is bounded. Then R is IR-bounded.*

Proof. For each function symbol f in Σ, consider the term $t_f = f(x_1, \ldots, x_m)$, where m is the arity of f and $x_1, \ldots, x_m$ are variables. Since R is bounded, there is an n_f such that for any normalized substitution θ, $\theta(t_f) \xrightarrow{\leq n_f}_R \theta(t_f)\downarrow$. Let u be a innermost redex with f as its root symbol. Note that there is a normalized substitution θ such that $\theta(t_f) = u$, and thus $u \xrightarrow{\leq n_f}_R u\downarrow$. Let n be the largest such n_f for any f in Σ. Then for any innermost redex u', $u' \xrightarrow{\leq n}_R u'\downarrow$. Therefore, R is IR-bounded. □

Lemma 15. *Suppose a convergent rewrite system R is IR-bounded. Then R is bounded.*

Proof. Since R is IR-bounded, there is a bound n such that for any innermost redex u, $u \xrightarrow{\leq n}_R u\downarrow$. Let t be a term, and θ be a normalized substitution. The set of positions where $\theta(t)$ could be rewritten is a subset of $\mathcal{FP}os(t)$. Consider a position p in $\mathcal{FP}os(t)$ such that $\theta(t)|_p$ is an innermost redex. Since R is IR-bounded, $\theta(t)|_p \xrightarrow{\leq n}_R (\theta(t)|_p)\downarrow$. Once $\theta(t)|_p$ is rewritten, the only subterms that can become new innermost redexes are its ancestors. Clearly then the entire term $\theta(t)$ can be rewritten in no more than $n \cdot |\mathcal{FP}os(t)|$ steps. Therefore R is bounded. □

With this result, we can easily show one direction of the equivalence.

Lemma 16. *Suppose a convergent rewrite system R has a finite forward closure $FC(R)$. Then R has the finite variant property.*

Proof. If $FC(R)$ is finite, then $FC(R) = FC_k(R)$ for some k. By Corollary 2, given an innermost redex t, $t \to_{FC(R)} t\downarrow$. So $t \to_{FC_k(R)} t\downarrow$, and by Lemma 10 there is a $k' \leq k+1$ such that $t \to_R^{k'} t\downarrow$. Therefore R is IR-bounded. By Lemma 15, R is bounded, and thus R has the finite variant property. □

In the other direction, things are a bit more complicated. We relate the variants of a rewrite system to redundancy. First, given a rewrite system R, we define the following set of rules, $\mathcal{V}_R$.

Definition 4. *For a convergent rewrite system R that has the finite variant property, we define*

$$\mathcal{V}_R = \{\theta(l) \to l' \mid l \to r \in R \textit{ and } (l', \theta) \in [\![l]\!]^\star \textit{ and } \theta(l) \textit{ is an innermost redex}\}$$

The rules in $\mathcal{V}_R$ correspond to variants of the left-hand sides of rules in R. The next three lemmas use this set to prove that a convergent system with the finite variant property has a finite forward closure.

Lemma 17. *Suppose a convergent rewrite system R has the finite variant property. Then there is a $k > 0$ such that each rule in $\mathcal{V}_R$ is redundant in $FC_k(R)$.*

Proof. Since R has the finite variant property, for any term t, $[\![l]\!]^{\star}$ is finite. Thus $\mathcal{V}_R$ is finite. For each $\theta(l) \to l'$ in $\mathcal{V}_R$, $\theta(l)$ is an innermost redex and l' is its normal form. Thus, by Lemma 13, there is a $k > 0$ such that $\theta(l) \to_{FC_k(R)} l'$. Let k' be the max of all such k. Each rule in $\mathcal{V}_R$ is redundant in $FC_{k'}(R)$. □

Lemma 18. *Suppose a convergent rewrite system R has the finite variant property, and let $k > 0$ be such that each rule in $\mathcal{V}_R$ is redundant in $FC_k(R)$. Then every innermost redex can be reduced to its normal form in one step modulo $FC_k(R)$.*

Proof. Let $\theta(l)$ be an innermost redex where l is the left-hand side of a rule in R. Let s be its normal form. Clearly the substitution θ has to be a normalized substitution (over $\mathcal{V}ar(l)$) for otherwise $\theta(l)$ would not be an innermost redex. Since R has the finite variant property, there is a variant (l', σ) of l such that $(s, \theta) \sqsubseteq (l', \sigma)$. Thus there is a substitution η such that $\theta = \eta \circ \sigma$ and $s = \eta(l')$. Thus, since $\sigma(l)$ is also an innermost redex, $\theta(l) \to s$ is an instance of the rule $\sigma(l) \to l' \in \mathcal{V}_R$. Since l' is the normal form of $\sigma(l)$, by Lemma 7, $\sigma(l) \to l'$ must not be strictly redundant in $FC_k(R)$. So $\sigma(l) \to l'$, and therefore $\theta(l) \to s$, must be an instance of a rule in $FC_k(R)$ and we are done. □

Lemma 19. *Suppose a convergent rewrite system R has the finite variant property. Then R has a finite forward closure $FC(R)$.*

We have now equated the finite variant property to the finiteness of forward closure. All the results in this section lead us to the following theorem.

Theorem 2. *Let R be a convergent rewrite system. The following statements are equivalent:*

(i) R is bounded.
(ii) R is IR-bounded.
(iii) R has a finite forward closure
(iv) R has the finite variant property

7 Undecidability of Finiteness of Forward Closure

We will prove the undecidability of the finiteness of forward closure by reduction from the uniform mortality problem for deterministic Turing machines [11]. Given a deterministic Turing machine M, the machine is said to be *uniformly mortal* if and only if there is a number k such that, for any instantaneous description I of M, the number of transitions that M can make starting from I is at most k.

We represent a deterministic Turing machine M as a tuple $(\Gamma, \flat, Q, \delta, F)$, where Γ is the tape alphabet, $\flat \in \Gamma$ is the blank symbol, Q is the set of states, $F \subset Q$ is the set of final states, and $\delta \colon (Q \setminus F) \times \Gamma \to Q \times \Gamma \times \{L, R\}$ is the transition function. We assume that $\Gamma \cap Q = \emptyset$.

An *instantaneous description* (ID) of M is represented as a tuple (u, q, γ, v), where u is a suffix of the string to the left of the tape head, q is the current state,

γ is the current symbol under the tape head, and v is a prefix of the string to the right of the head. The strings to the left and right of the tape head may be infinite, but only a finite suffix and prefix, respectively, will contain non-blank symbols. Therefore, we let u be the longest suffix of the string to the left of the tape head such that $u \neq \text{ƀ}u'$. Similarly, v is the longest prefix of the string to the right of the head such that $v \neq v'\text{ƀ}$.

For IDs I_1 and I_2 of M, $I_1 \vdash I_2$ if and only if there is a transition in δ that would move M from I_1 to I_2. Note that this usage of $\vdash$ is separate from the usual meaning of "proves". An ID $I = (u, q, \gamma, v)$ is *final* if and only if $q \in F$.

The notion of an ID can be extended to that of a *window*. A window W of M is a tuple (u, q, γ, v) such that $u \in \text{ƀ}^* u'$ and $v \in v'\text{ƀ}^*$ for some u' and v' such that $I = (u', q, \gamma, v')$ is an ID of M. In this case, W *extends* I. The width of W is $|W| = |u| + |v| + 1$. For windows W_1 and W_2, $W_1 \vdash W_2$ if and only if $|W_1| = |W_2|$ and there are IDs I_1 and I_2 such that W_1 and W_2 extend I_1 and I_2, respectively, and $I_1 \vdash I_2$.

Proposition 3. *Let M be a Turing machine, and let $I_1, I_2, \ldots, I_n$ be IDs of M such that $I_1 \vdash I_2 \vdash \cdots \vdash I_n$. Then there is a width k and windows $W_1, W_2, \ldots, W_n$, each with width k, such that each W_i extends I_i and $W_1 \vdash W_2 \vdash \cdots \vdash W_n$.*

For any given Turing machine M, we construct a rewrite system R_M and show that M is uniformly mortal if and only if $FC(R_M)$ is finite. Our system is over the signature $\Sigma = Q \cup \Gamma \cup \{\epsilon, s\}$, where each $q \in Q$ has arity 3, each $\gamma \in \Gamma$ has arity 1, ϵ is a constant, and s has arity 1. We assume an infinite set $\mathcal{X}$ of variables.

We can encode a number n as a term $s^n(\epsilon)$. Each ternary function symbol $q \in Q$ represents a window in state q, and each monadic function symbol $\gamma \in \Gamma$ represents concatenation on the left by that symbol. We encode a string $w = \gamma_1 \cdots \gamma_n$ over Γ as a term $enc(w) = (\gamma_1 \circ \cdots \circ \gamma_n)(\epsilon)$, where $\circ$ is function composition (i.e., $(f \circ g)(x) = f(g(x))$). We can then encode a window (u, q, γ, v) as a term $q(enc(u^{\text{rev}}), \gamma(enc(v)), s^n(\epsilon))$, where u^{rev} is the reverse of the string u, and n is the number of transitions the machine is allowed to make.

We say two terms t_1 and t_2 are *sequential* if and only if t_1 and t_2 both have root symbols from Q and $t_1|_3 = s(t_2|_3)$. We say a term t is *legal* if and only if there is a window W of M such that t encodes W. We say a term is *illegal* if and only if it has a root symbol from Q but is not legal.

Definition 5. *We define a function $\phi\colon T(\Sigma, \mathcal{X}) \to T(\Sigma)$ to transform illegal terms into legal terms. For all $q \in Q$,*

$$\phi(q(t_1, t_2, t_3)) = q(\phi'_\Gamma(t_1),\ \phi'_\Gamma(t_2),\ \phi'_{\{s\}}(t_3))$$

where $\phi'_S\colon T(\Sigma, \mathcal{X}) \to T(\Sigma)$ is a helper function parameterized by a signature $S \subseteq \Sigma$,

$$\phi'_S(t) = \begin{cases} f(\phi'_S(t')) & \textit{if } t = f(t') \textit{ for some } f^{(1)} \in S \\ \epsilon & \textit{otherwise} \end{cases}$$

The function ϕ'_S finds the “highest” occurrence of a term whose root symbol does not belong in a string over signature S and replaces it with ϵ. The function ϕ uses this to ensure that subterms encode valid tape strings (over the signature Γ) or numbers (over the signature $\{s\}$).

We can now construct our rewrite system R_M from a machine M.

Definition 6. *Let $M = (\Gamma, b\!\!\!/, Q, \delta, F)$ be a deterministic Turing machine. First set $R_M := \emptyset$. For each left-moving transition $(q, \gamma) \mapsto (q', \gamma', L)$ in δ, extend R_M by*

$$R_M := R_M \cup \{q(\gamma_0(x),\, \gamma(y),\, s(z)) \to q'(x,\, \gamma_0(\gamma'(y)),\, z) \mid \gamma_0 \in \Gamma\}$$

where x, y, and z are variables. Then, for each right-moving transition $(q, \gamma) \mapsto (q', \gamma', R)$ in δ, extend R_M by

$$R_M := R_M \cup \{q(x,\, \gamma(\gamma_0(y)),\, s(z)) \to q'(\gamma'(x),\, \gamma_0(y),\, z) \mid \gamma_0 \in \Gamma\}$$

where again, x, y, and z are variables.

We first prove some basic properties of the rewrite system R_M.

Lemma 20. *Let M be a deterministic Turing machine, let t_1 be an innermost redex, and let t_2 and t_3 be terms such that $t_1 \to_{R_M} t_2$ and $t_1 \to_{R_M} t_3$. Then $t_2 = t_3$.*

Lemma 21. *Let M be a deterministic Turing machine. Then the rewrite system R_M is convergent.*

Lemma 22. *Let M be a deterministic Turing machine, let t_1 be an innermost redex, and let t_2 be a term such that $t_1 \to_{R_M} t_2$. Then t_2 is either an innermost redex or in normal form.*

Lemma 23. *Let M be a deterministic Turing machine, let t_1 be an innermost redex, and let t_2 be a term such that $t_1 \to_{R_M} t_2$. Then t_1 and t_2 are sequential.*

Our goal in this section is to show that the rewrite system R_M models computation of the machine M. Unfortunately, there are terms over Σ that are R_M-reducible but do not encode any window of M. With the ϕ function, we can map such illegal terms to a representative legal term. The following lemma shows that ϕ preserves the R_M-reducibility of the term, and thus we can focus our attention on legal terms.

Lemma 24. *Let M be a deterministic Turing machine, and let t_1 be an innermost redex and t_2 be a term such that t_1 and t_2 have root symbols from Q. Then $t_1 \to^k_{R_M} t_2$ for some $k > 0$ if and only if $\phi(t_1) \to^k_{R_M} \phi(t_2)$.*

Proof (Sketch). The idea is that ϕ'_Γ and $\phi'_{\{s\}}$ can be pushed below the subterms of instances of rules in R_M. So if $t_1 \to^k_{R_M} t_2$, then for any step $t \to t'$, there is a rule $l \to r$ in R_M such that $t \to t' \sqsubseteq_\sigma l \to r$. If we apply ϕ, the ϕ'_Γ and $\phi'_{\{s\}}$ will be pushed down into $\sigma(x)$ for each $x \in \mathcal{V}ar(l)$, and thus $\phi(t) \to \phi(t')$. Therefore we have $\phi(t_1) \to^k_{R_M} \phi(t_2)$.

Conversely, if $t_1 \not\to^k_{R_M} t_2$, then applying ϕ cannot fix things, because it only changes things below the rule. Therefore $\phi(t_1) \not\to^k_{R_M} \phi(t_2)$. □

Corollary 3. *Let M be a deterministic Turing machine, and let t be a term with a root symbol from Q such that no proper subterm of t is reducible. Then $\phi(t)$ is in R_M-normal form if and only if t is in R_M-normal form.*

Now we can relate transitions between windows of M to rewriting terms that encode them in R_M.

Lemma 25. *Let M be a deterministic Turing machine, let W_1 and W_2 be windows of M with equal width, and let t_1 and t_2 be sequential terms encoding W_1 and W_2, respectively. Then $W_1 \vdash W_2$ if and only if $t_1 \to_{R_M} t_2$.*

Proof (Sketch). Here the idea is that if t_1 and t_2 encode W_1 and W_2, respectively, and if there is a transition from W_1 to W_2, then it corresponds to a unique rule in R_M that rewrites t_1 to t_2. Similarly, if there is a rule that rewrites t_1 to t_2, it corresponds to a unique transition from W_1 to W_2. □

Lemma 26. *Let M be a deterministic Turing machine, let W be a window of M, and let t be a term encoding W. If W is final, then t is in normal form.*

Lemma 27. *Let M be a deterministic Turing machine. Then M is uniformly mortal if and only if the rewrite system R_M is IR-bounded.*

Proof (Sketch). We first show a one-to-one correspondence between windows of M and legal terms. Transitions between windows correspond to rewrites in R_M. If the machine is uniformly mortal, the bound corresponds to IR-boundedness. Otherwise there exists some unbounded rewrite sequence starting from an innermost redex. □

Theorem 3. *It is undecidable to check, given a finite convergent term rewriting system, whether it has a finite forward closure.*

Proof. By Lemma 27, we have reduced the uniform mortality problem for deterministic Turing machines to the IR-boundedness problem. Therefore, by Theorem 2, the uniform mortality problem can be reduced to checking if R has a finite forward closure. By Lemma 21, for any deterministic Turing machine M we know that R_M is convergent. Thus it is undecidable whether a finite convergent term rewriting system has a finite forward closure. □

Corollary 4. *It is undecidable to check, given a finite convergent term rewriting system, whether it has the finite variant property.*

8 Modularity of Forward Closure

In this section we examine how forward closure behaves when rewrite systems are combined. We first consider the modularity of the finiteness of forward closure, i.e., whether the property is preserved when combining systems over disjoint signatures.

Theorem 4. *Let R_1 and R_2 be finite rewrite systems over signatures Σ_1 and Σ_2 respectively. If $\Sigma_1 \cap \Sigma_2 = \emptyset$, then $FC(R_1 \cup R_2) = FC(R_1) \cup FC(R_2)$.*

Proof. Suppose $FC(R_1 \cup R_2) \supsetneq FC(R_1) \cup FC(R_2)$. Then there must be a k such that either a rule from $FC_k(R_1)$ was overlapped with a rule from R_2, or a rule from $FC_k(R_2)$ with a rule from R_1. We will assume the former without loss of generality. Thus there is a rule $l \to r$ in $FC(R_1 \cup R_2)$ such that

$$(l \to r) = (l_1 \to r_1) \rightsquigarrow_p (l_2 \to r_2)$$

where $p \in \mathcal{FP}os(r_1)$, $(l_1 \to r_1) \in FC_k(R_1)$, and $(l_2 \to r_2) \in R_2$. So then

$$(l \to r) = \theta(l_1) \to \theta(r_1[r_2]_p)$$

where $\theta = mgu(r_1|_p =^? l_2)$. However, since Σ_1 and Σ_2 are disjoint, and since p is a non-variable position in r_1, the terms $r_1|_p$ and l_2 are not unifiable due to function clash. This is a contradiction. Since $FC(R_1 \cup R_2) \supseteq FC(R_1) \cup FC(R_2)$, we have that $FC(R_1 \cup R_2) = FC(R_1) \cup FC(R_2)$. □

However, if the systems are allowed to share constants, then even if the systems have finite forward closures their union may not.

Example 2. Let $R_1 = \{f(a, h(x)) \to h(f(b, x))\}$, and let $R_2 = \{b \to a\}$, where a and b are constants. These systems are clearly convergent and forward-closed. However, consider their union,

$$R_1 \cup R_2 = \{f(a, h(x)) \to h(f(b, x)),\ b \to a\}$$

This system is convergent. However, it has an infinite forward closure, because for all $k > 0$:

$$NR_{2k}(R_1 \cup R_2) = \{f(a, h^{k+1}(x)) \to h^{k+1}(f(a, x))\}$$

This is obtained by overlapping the rule from $NR_{2k-2}(R_1 \cup R_2)$ first with the rule from R_2, then with the rule from R_1 (this is why the rules occur in every *other* step of forward closure). None of these rules are redundant, because they are not instances of existing rules and the ground instances obtained by applying the substitution $\{x \mapsto a\}$ cannot be proven by smaller instances of rules. Since $NR_k(R_1 \cup R_2) \neq \emptyset$ for any k, by Lemma 9, $FC(R_1 \cup R_2)$ is not finite. □

9 Relationship to Runtime Complexity

Inspired by a comment from one of our reviewers, we examined the relationship to the field of *runtime complexity*, as described in [12]. The notion of the runtime complexity of a rewrite system is similar to the IR-boundedness property. However, while runtime complexity gives a bound for all rewrite sequence from an innermost redex, IR-boundedness only guarantees that a rewrite sequence *exists* which is shorter than the bound. For this reason, a rewrite system with $O(1)$ runtime complexity is IR-bounded, but it seems that the inverse is not necessarily true.

Several tools exist for automatically checking the runtime complexity of a rewrite system, such as CaT[3] and TCT[4]. These tools can now be used to recognize a class of rewrite systems with the finite variant property.

10 Conclusion and Future Work

Inspired by Basic Syntactic Mutation [5, 14], we explored forward closure and its relation to the finite variant property [6]. We found that, with suitable redundancy constraints, the finiteness of forward closure is equivalent to the finite variant property. We also showed that finiteness of forward closure is undecidable, even for convergent rewrite systems.

A great deal of research has gone into finding ways to decide if a rewrite system has the finite variant property [8]. As we have shown the equivalence of the finite variant property and finiteness of forward closure, we have a convenient procedure for checking the finite variant property, much like Knuth-Bendix completion provides a procedure for deciding the word problem [13]. As the finiteness of forward closure is undecidable, the procedure may not terminate, but if the rewrite system has the finite variant property, the procedure will terminate in a finite number of steps.

Our future work centers around extending forward closure to work modulo equational theories. The most important is the theory of AC (associativity and commutativity), which has many practical applications, but we hope to consider a much more general class of theories. We will also examine in more detail how forward closure behaves when rewrite systems are combined that are not completely disjoint.

Acknowledgements. We wish to thank Serdar Erbatur, Ralf Sasse, and the reviewers for their detailed comments, style suggestions, and proof corrections which helped considerably to improve this paper.

References

[1] Anantharaman, S., Erbatur, S., Lynch, C., Narendran, P., Rusinowitch, M.: Unification Modulo Synchronous Distributivity. In: Gramlich, B., Miller, D., Sattler, U. (eds.) IJCAR 2012. LNCS, vol. 7364, pp. 14–29. Springer, Heidelberg (2012)

[3] `http://cl-informatik.uibk.ac.at/software/cat/`

[4] `http://cl-informatik.uibk.ac.at/software/tct/`

[2] Baader, F., Nipkow, T.: Term Rewriting and All That. Cambridge University Press (1999)

[3] Baader, F., Snyder, W.: Unification Theory. In: Robinson, J.A., Voronkov, A. (eds.) Handbook of Automated Reasoning, pp. 440–526. Elsevier Science Publishers BV (1999)

[4] Bouchard, C., Gero, K.A., Lynch, C., Narendran, P.: On Forward Closure and the Finite Variant Property. Technical report, Dept. of Computer Science, University at Albany—SUNY (July 2013)

[5] Bouchard, C., Gero, K.A., Narendran, P.: Some Notes on Basic Syntactic Mutation. In: Escobar, S., Korovin, K., Rybakov, V. (eds.) Proceedings 26th International Workshop on Unification, pp. 9–14 (2012)

[6] Comon-Lundh, H., Delaune, S.: The finite variant property: How to get rid of some algebraic properties. In: Giesl, J. (ed.) RTA 2005. LNCS, vol. 3467, pp. 294–307. Springer, Heidelberg (2005)

[7] Erbatur, S., Escobar, S., Kapur, D., Liu, Z., Lynch, C., Meadows, C., Meseguer, J., Narendran, P., Santiago, S., Sasse, R.: Effective Symbolic Protocol Analysis via Equational Irreducibility Conditions. In: Foresti, S., Yung, M., Martinelli, F. (eds.) ESORICS 2012. LNCS, vol. 7459, pp. 73–90. Springer, Heidelberg (2012)

[8] Escobar, S., Meseguer, J., Sasse, R.: Effectively checking the finite variant property. In: Voronkov, A. (ed.) RTA 2008. LNCS, vol. 5117, pp. 79–93. Springer, Heidelberg (2008)

[9] Escobar, S., Sasse, R., Meseguer, J.: Folding variant narrowing and optimal variant termination. Journal of Logic and Algebraic Programming 81(7-8), 898–928 (2012); Rewriting Logic and its Applications

[10] Hermann, M.: Chain properties of rule closures. Formal Aspects of Computing 2(1), 207–225 (1990)

[11] Hillebrand, G.G., Kanellakis, P.C., Mairson, H.G., Vardi, M.Y.: Undecidable Boundedness Problems for Datalog Programs. Journal of Logic Programming 25(2), 163–190 (1995)

[12] Hirokawa, N., Moser, G.: Automated Complexity Analysis Based on the Dependency Pair Method. In: Armando, A., Baumgartner, P., Dowek, G. (eds.) IJCAR 2008. LNCS (LNAI), vol. 5195, pp. 364–379. Springer, Heidelberg (2008)

[13] Knuth, D.E., Bendix, P.: Simple word problems in universal algebras. In: Leech, J. (ed.) Computational Problems in Abstract Algebra, pp. 263–297. Pergamon Press (1970)

[14] Lynch, C., Morawska, B.: Basic Syntactic Mutation. In: Voronkov, A. (ed.) CADE 2002. LNCS (LNAI), vol. 2392, pp. 471–485. Springer, Heidelberg (2002)

[15] Nieuwenhuis, R., Rubio, A.: Paramodulation-Based Theorem Proving. In: Robinson, J.A., Voronkov, A. (eds.) Handbook of Automated Reasoning, pp. 371–443. Elsevier, MIT Press (2001)