Consistency and Invariants — Extended Abstract

Stefan Kahrs

School of Computing
University of Kent
Canterbury, United Kingdom
S.M.Kahrs@kent.ac.uk

Abstract

We are looking at the specific problem of proving consistency of TRSs and the more general problem of proving program invariants. On the specific side we show that almost non-ω-overlapping constructor TRSs are consistent (and have unique normal forms), but the proof technique to get there shows us ways how to tackle the much more general problem of proving program invariants. To prove invariants we are using *proof graphs*, union/find-structures that carry proof information. As index set for these structures we use finite Σ-coalgebras — this is sufficient, because any equational proof is a finite object.

1 Introduction

Note: this is based on a paper that appeared in FSCD 2016 [7].

For over 40 years [10] it has been known that TRSs that are left-linear and non-overlapping are confluent, and for over 30 years [6] that non-overlapping on its own may not even give us unique normal forms:

Example 1. *By Huet [6]:* $\{F(x,x) \to A, F(x,G(x)) \to B, C \to G(C)\}$. *The term* $F(C,C)$ *possesses two distinct normal forms,* A *and* B.

Moreover, if we replace the right-hand side of the first rule with x then the TRS is even inconsistent. The first two rules overlap syntactically, in a certain sense: if we can instantiate x with the infinite term $G(G(\cdots))$ then the first two rules would both be redexes, i.e. the rules ω-overlap. This creates the question: *do non-ω-overlapping TRSs have unique normal forms*? This was first conjectured in the 1980s by Ogawa [9]. Not only can UN be replaced by CON, we can simplify the problem from TRSs to constructor TRSs, because there is a translation between such systems [7] which preserves and reflects consistency, and which (modulo some minor issues) also preserves the non-ω-overlapping property.

Properties such as confluence, uniqueness of normal forms, and consistency are examples of program invariants, properties that are implied by $t =_R u$. Or rather: a confluence proof shows that the joinability relation $\downarrow$ is a program invariant: if $t =_R u$ then $t \downarrow u$. Of all program invariants, consistency is the bottom line: if $t =_R u$ is vacuously true then we cannot deduce anything from it. Therefore, whenever we prove a non-vacuous program invariant, we should get a proof of consistency for free.

In general, whether a property is an invariant is undecidable for TRSs, so we are looking for classes of TRSs for which the problem is easier, and sometimes for over-approximations of our invariants. What we typically have as an *invariant candidate* is an inductively defined relation which is in general not a congruence relation, but might be for well-behaved TRSs. Moreover, the missing ingredient is often transitivity, e.g. if $\downarrow$ is transitive then it is also a congruence relation.

How does one show that a certain inductively defined invariant candidate $\triangle = \mu x.F(x)$ is indeed an invariant, i.e. in particular that it is transitive? Instead of considering whether

B. Accatoli and B. Felgenhauer (eds.); IWC 2017, pp. 1–6

$t =_R u$ or $t \triangle u$ do or do not hold *in general* (typically undecidable properties), we consider whether we can prove them if we are only allowed to use in the proofs terms from some finite set A — more specifically, we want that set to be closed under subterms, which means that the set is a coalgebra of the signature functor. For this finitist argument to work, we need that F is continuous, that we never need an infinite part of the relation $\triangle$ to deduce $t \triangle u$. The condition "only terms from A are allowed" can typically be achieved by defining $\triangle_A = \mu x.id_A \cdot F(x) \cdot id_A$, where id_A is the identity on that set A.

If that relation $\triangle_A$ is always transitive we must have that $\triangle$ itself is always transitive. Even if that were not the case it would suffice if we could find, for any finite coalgebra A, a coalgebra $B \supset A$, and an equivalence relation $=_B$ such that $\triangle_A \subseteq =_B \subseteq \triangle_B$.

2 Proof Graphs

One way of showing that a (finite) relation $\triangle$ is transitive is to build a complete finite representation of it that necessarily implies the transitivity of the relation. For this purpose we are using union/find-structures [5, 12]. Union/find-structures are generally used to represent finite equivalence relations in an efficient way; in this context their efficiency is not a particular concern though it is related to the relative simplicity of the manipulation of the equivalence. A special case of proof graph to support confluence proofs had been used by Sakai et.al.[11].

What we are looking for are union/find-structures ϱ whose associate equivalence relations $=_\varrho$ are necessarily subrelations of our relation $\triangle$. For this to get off the ground we need at least that $\triangle$ is reflexive and symmetric.

Definition 1. *An* equivalence candidate *is a reflexive and symmetric relation.*

Secondly, to achieve the goal $=_\varrho \subseteq \triangle$ we need that the edges used in our structure have sufficiently "nice" properties. Ideally, this means that they are "prefixes":

Definition 2. *A* prefix *of an equivalence candidate $\triangle$ is a relation $\lhd$ such that $\lhd \cdot \triangle \subseteq \triangle$.*

It is easy to see that prefixes of $\triangle$ have various nice algebraic properties: they are closed under arbitrary union, composition, and subrelations. A union/find-structure ϱ is then a proof graph for $\triangle$ if its edge relation $\xrightarrow{\varrho}$ is a prefix of $\triangle$. This suffices to imply $=_\varrho \subseteq \triangle$.

Notice that if $\triangle$ is transitive then it is itself its own prefix. However, (i) we do not know that to begin with but we may know that about a relation $\lhd$; (ii) if we constrain our edges to use a particular kind of prefix then we can also exploit that knowledge.

The idea behind the notion of proof graph is that our equivalence candidates are inductively defined relations that trivially have certain relations as prefixes, as part of their definition.

Sometimes, we need to prove the transitivity of a candidate $\triangle_A$ indirectly, through a proof graph for $\triangle_B$. This works as follows:

Definition 3. *Let (A, α) and (B, β) be proof graphs for candidates $\triangle_A$ and $\triangle_B$, respectively. Then a function $f : A \to B$ is a* proof graph morphism *iff we have the following properties:*

- $\forall x, y \in A.\, x \xrightarrow{\alpha} y \Rightarrow f(x) \xrightarrow{\beta} f(y)$
- $\forall x, y \in A.\, f(x) \triangle_B f(y) \Rightarrow x \triangle_A y$

Clearly, proof graph morphisms can be composed and thus form a category. Moreover, each proof graph morphism f induces an equivalence on its domain A: $x =_f y \iff f(x) =_\beta f(y)$, and by the second condition this equivalence must be a subrelation of $\triangle_A$.

Generally, we have two typical ways of creating proof graph morphisms: (i) by adding edges to the graph, without changing the candidate or index set, and (ii) by embedding A into a larger set B, preserving all edges, but constraining the relation $\triangle_B$ suitably. For example, we may want to do this for a confluence proof by providing in B missing common reducts whilst restricting root rewrite steps to reside within A.

One thing we can do with proof graphs is add prefix edges to a normal forms. If we want to limit ourselves to edges of a particular prefix $\lhd$ we can always extend a given proof graph (via proof graph morphisms), so that the resulting proof graph ω has the UN property for proof graphs:

Definition 4. *A proof graph ω has the* UN property for prefix $\lhd$ *iff*

$$\forall t, s.\ (\neg(\exists u.\ t \xrightarrow{\omega} u)) \Rightarrow t \lhd^* s \Rightarrow t =_\omega s$$

.

The reason this is called a "UN property", because if two normal forms ("roots" in union/find-structure terminology) of the proof graph have a common $\lhd$ reduct then they must be the same.

Lemma 1. *Any proof graph has an extension that has the UN property for any prefix of its equivalence candidate.*

Compared to considering the set of all terms we gain additional induction principles when organising a set of terms as index set for a proof graph ω — as we limit ourselves to finitely many terms, finitely many equivalence classes, and in particular: a terminating edge relation $\xrightarrow{\omega}$.

3 Term-Coalgebras, and relations on them

Relations on terms can more generally be viewed as relations on or between Σ-coalgebras. This can be useful to stratify the reasoning on terms, one finite Σ-coalgebra at a time.

In order to consider coalgebras of signatures Σ we would have to view signatures as functors on the category Set. However, we only need here the following special instance of this concept:

Definition 5. *Given a signature Σ, a* term-coalgebra *is a set $A \subseteq Ter^\infty(\Sigma, \emptyset)$ which is closed under subterms. It is called* finite *if it is a finite set, and* strongly finite *if in addition $A \subseteq Ter(\Sigma, \emptyset)$.*

More generally, Σ-coalgebras A would be characterized by a function $\upsilon : A \to \Sigma(A)$ which maps a node to a structure containing its root function symbol and the list of its subnodes. We also allow for variables in term-coalgebras by "freezing" them, i.e. when considered as a member of a term-coalgebra a variable is a nullary constructor. For heterogeneous relations between term-coalgebras we must therefore have that the variable set X is the same, so that they are coalgebras of the same functor.

One ingredient to define relations between or on term-coalgebras for a signature Σ we use the following notation: if $R \subseteq A \times B$, where A and B are term-coalgebras A and B then $\widetilde{R} \subseteq A \times B$ is defined as follows:

$$\begin{aligned}\forall t \in A.\, \forall u \in B.\, t\ \widetilde{R}\ u \iff \exists F \in \Sigma.\, \exists a_1, \ldots, a_n \in A.\, \exists b_1, \ldots, b_n \in B.\\ t = F(a_1, \ldots, a_n) \wedge u = F(b_1, \ldots, b_n) \wedge \forall i.\, a_i\ R\ b_i\end{aligned}$$

This concept was first used in [3, 4] (though with added reflexivity). For constructur signatures, we use the notations $\overline{R}$ and $\widehat{R}$ to mean $\widetilde{R}$ for the subsignatures Σ_d and Σ_c, respectively. In particular, $t\ \widehat{id}\ t$ iff the root symbol of t is a constructor, and so $\widehat{R} \cdot \overline{S} = \emptyset$. We still use $\widetilde{R}$ for constructor signatures, to refer to the combined signature; hence $\widetilde{R} = \overline{R} \cup \widehat{R}$.

If id_V is the coreflexive identity on variables then we can express consistency of a relation R relation-algebraically as $id_V \cdot R \cdot id_V \subseteq id_V$. However, we already have consistency issues when a relation R relates any terms topped with distinct constructors. Relations that do not do that we call "constructor-consistent": $\widehat{id} \cdot R \cdot \widehat{id} \subseteq \widehat{1}$, where "1" is top element of the lattice of relations. To reason about pattern matching we need something even stronger than that:

Definition 6. *A relation R between term-coalgebras is called* constructor-compatible *iff*

$$\widehat{id} \cdot R \cdot \widehat{id} \subseteq \widehat{R}.$$

Constructor-compatible relations are preserved by arbitrary union; consequently, relations defined as $\mu x.\, f(x)$ are constructor-compatible whenever the function f preserves this property. The same applies to consistent or constructor-consistent relations.

For "well-behaved" Constructor TRSs we would expect these forms of consistency to be properties of the standard rewrite congruence $=_R$. However, constructor-compatibility can sometimes be lacking in strange ways:

Example 2. $F(x,x) \to x, F(y, G(G(y))) \to G(G(G(y))), A \to G(G(A))$ *These rules are almost non-ω-overlapping.*

In Example 2, we have $G(G(A)) =_R G(G(G(A)))$ but we do not have $G(A) =_R G(G(A))$, i.e. $=_R$ is not constructor-compatible. The issue goes away (globally) if we request that for all constructors there are matching projection operations, e.g. here we would need a symbol π_G with rewrite rule $\pi_G(G(x)) \to x$; locally (at finite term-coalgebras) the issue would persist unless we ruled out term-coalgebras that contained $G(t)$ without containing $\pi_G(G(t))$. Alternatively, we can use an equivalence which over-approximates $=_R$. Any invariant of such an over-approximation must be an invariant of the original relation.

4 Redex Behaviour

Definition 7. *A relation R between term-coalgebras is called* Σ-closed *iff $\widetilde{R} \subseteq R$.*

Note: this is standard terminology taken from [1], except that we generalise it to coalgebras. When dealing with almost non-ω-overlapping constructor-rewriting, we sometimes require a stronger form of Σ-closure, because trivial ω-overlaps can cause anomalies, as we have seen in Example 2.

Definition 8. *An equivalence relation $=_\varrho$ on a term-coalgebra A is called* strongly Σ_c-closed *iff $\nu x.\, =_\varrho \cdot \widehat{x} \cup \widehat{x} \cdot =_\varrho \subseteq =_\varrho$.*

Notice that the construction in the definition uses a largest fixpoint, which means that even congruence relations may not have this property. For instance, let $=_\varrho$ be the strong Σ_c-closure of $=_R$ of Example 2, then we now do have $G(A) =_\varrho G(G(A))$ and $A =_\varrho G(A)$.

Proposition 1. *Let $a \to b$ and $c \to d$ be two rewrite rules (of some constructor TRS) with only trivial ω-overlaps. Let $=_S$ be a constructor-compatible and Σ-closed equivalence. Then $\sigma(a) \equiv_{\overline{S}} \theta(c)$ implies $\sigma(b) =_S \theta(d)$.*

The reason this is true is that (i) every equation derived via the ω-unification algorithm still holds in $=_S$, (ii) Σ-closed equivalences "are" algebras, so that we can "interpret" eventually the anti-unifiers of b and d in $=_S$-equivalent environments, giving $=_S$-equivalent results.

5 An invariant

Given a TRS with signature Σ, and a term-coalgebra A, the relation $\Downarrow_A$ is a relation on A defined as follows:

$$\Downarrow_A \doteq \mu x.\, \nu y.\, (id_A \cdot (\overline{x} \cup \widehat{y}) \cdot id_A)^* \cdot (\xrightarrow{\epsilon}\cdot\, x \cup id \cup x \cdot\xleftarrow{\epsilon}) \cdot (id_A \cdot (\overline{x} \cup \widehat{y}) \cdot id_A)^*$$

The reason for the largest fixpoint in the definition are almost-non-ω-overlapping systems such as Example 2, i.e. we intend to use $\Downarrow_A$ as an invariant for these systems. The relation $\Downarrow$ bears some similarity to joinability, but has one stronger feature: we have that $\widetilde{\Downarrow}$ is a prefix of $\Downarrow$. Unlike joinability, $\Downarrow$ is not constructor-compatible (in general), but it is constructor-consistent. One can define similar invariants [7] that are constructor-compatible too, but these fit less well with our general notion of proof graph, in particular we would for those relations not have the corresponding property that $\widehat{\Downarrow_A}$ is a prefix of $\Downarrow_A$.

6 Building a Universal Proof Graph

To show that our proposed invariant relation $\Downarrow_A$ is indeed an invariant for almost non=ω-overlapping constructor TRSs we can show instead that it is transitive — because the universal properties of $=_R$ would imply the rest. For this purpose we need to build a *universal proof graph* for $\Downarrow_A$. This is a proof graph ϱ on A such that $=_\varrho$ and $\Downarrow_A$ coincide.

In general, this kind of construction might be indirect via a proof graph morphism f, i.e. through a proof graph for $\Downarrow_B$ we would get that $=_f$ coincides with $\Downarrow_A$, not the proof graph equivalences themselves. For consistency proofs we can typically stick to one coalgebra and one invariant.

Definition 9. *We build a proof graph ϱ for $\Downarrow_A$ as follows:*

- *Form PG α whose equivalence contains $id_A \cdot\xrightarrow{\epsilon}\cdot\, id_A$.*
- *Form extension β of this PG which has UN property w.r.t. prefix $\overline{\Downarrow_A}$.*
- *Form PG ϱ whose equivalence is the strong Σ_c-closure of $=_\beta$.*

Notice that we can form α whenever a TRS is non-overlapping (even when it is ω-overlapping). We can form β by Lemma 1; the knowledge of what the relation $\overline{\Downarrow_A}$ actually is, is a separate issue — it would typically require a separate data structure. Because the equivalence relation $\overline{\Downarrow_A}^*$ is a prefix too one can use a separate union/find structure for this purpose. Finally, the strong Σ_c-closure can always be performed on proof graphs of this relation, regardless of TRS.

For any inductively defined relation $\mu x.F(x)$ to prove that it coincides with $=_\varrho$ means to show that the inclusion $F(=_\varrho) \subseteq =_\varrho$ holds, i.e. $\forall t, u \in A.t\ F(=_\varrho\ u \Rightarrow t =_\varrho u$. If the equivalence $=_\varrho$ is derived from a proof graph, then we can show the property by induction on $\xrightarrow{\varrho}$. First, we define a well-founded ordering on pairs of terms from A:

$$(t, u) > (t', u') \iff t \xrightarrow{\varrho}{}^{+} t' \wedge u \xrightarrow{\varrho}{}^{*} u' \vee t \xrightarrow{\varrho}{}^{*} t' \wedge u \xrightarrow{\varrho}{}^{+} u'$$

Then we prove our goal by induction on this ordering, i.e. we are showing:

$$\forall t, u \in A.\, (\forall t', u' \in A.\, (t, u) > (t', u') \wedge t' \Downarrow_A u' \Rightarrow t' =_\varrho u') \Rightarrow (t\ F(=_\varrho)\ u \Rightarrow t =_\varrho u)$$

Proposition 2. *The proof graph ϱ is universal for $\Downarrow_A$ whenever the TRS is almost non-ω-overlapping.*

Theorem 1. *Almost non-ω-overlapping Constructor TRSs have a constructor-consistent equational theory.*

References

[1] Franz Baader and Tobias Nipkow. *Term Rewriting and All That.* Cambridge University Press, 1998.

[2] Hendrik P. Barendregt. *The Lambda-Calculus, its Syntax and Semantics.* North-Holland, Amsterdam, 1984.

[3] Jörg Endrullis, Helle Hvid Hansen, Dimitri Hendriks, Andrew Polonsky, and Alexandra Silva. A coinductive treatment of infinitary rewriting. In *Workshop on Infinitary Rewriting*, page 8, 2013.

[4] Jörg Endrullis, Dimitri Hendriks, Helle Hvid Hansen, Andrew Polonsky, and Alexandra Silva. A coinductive framework for infinitary rewriting and equational reasoning. In *Rewriting Techniques and Applications*, 2015.

[5] Bernard A. Galler and Michael J. Fisher. An improved equivalence algorithm. *Commun. ACM*, 7(5):301–303, May 1964.

[6] Gérard Huet. Confluent reductions: Abstract properties and applications to term rewriting systems. *Journal of the ACM*, 27:797–821, 1980.

[7] Stefan Kahrs and Connor Smith. Non-Omega-Overlapping TRSs are UN. In Delia Kesner and Brigitte Pientka, editors, *1st International Conference on Formal Structures for Computation and Deduction (FSCD 2016)*, volume 52 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 22:1–22:17, Dagstuhl, Germany, 2016. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[8] Jan Willem Klop. *Combinatory Reduction Systems.* PhD thesis, Centrum voor Wiskunde en Informatica, 1980.

[9] Mizuhito Ogawa and Satoshi Ono. On the uniquely converging property of nonlinear term rewriting systems. Technical Report IEICE COMP89-7, NTT Software Laboratories, 1989.

[10] Barry K. Rosen. Tree-manipulating systems and Church-Rosser theorems. *Journal of the ACM*, 20:160–187, 1973.

[11] Masahiko Sakai, Michio Oyamaguchi, and Mizuhito Ogawa. Non-e-overlapping, weakly shallow, and non-collapsing trss are confluent. In Amy P. Felty and Aart Middeldorp, editors, *Automated Deduction - CADE-25 - 25th International Conference on Automated Deduction, Berlin, Germany, August 1-7, 2015, Proceedings*, volume 9195 of *Lecture Notes in Computer Science*, pages 111–126. Springer, 2015.

[12] Robert Endre Tarjan. Efficiency of a good but not linear set union algorithm. *J. ACM*, 22(2):215–225, April 1975.