Non-$\omega$-overlappings TRSs are UN

Stefan Kahrs and Connor Smith
University of Kent
This is about:

• When is the equational theory of a TRS consistent (CON), when does it have unique normal forms (UN),...

• How can we prove it?

• How are these issues related to $\omega$-substitutions, substitutions with infinitary terms in their range ...

Ultimately, we got:

• The result mentioned in the title.

• A new proof technique for proving consistency.
The issue:

• We are generally interested in the consistency of equational theories, i.e. that not everything is equal to everything (on open terms)

• This is normally proved via confluence (CR): A confluent system is consistent, because:
  • Variables are normal forms
  • Distinct variables are distinct normal forms

• Other kinds of consistency proofs?
  • One can create a non-trivial equational model, but...
  • ...that is hard to “bootstrap” in this case (no CPO structure).
Standard TRS confluence criteria

• For terminating systems:
  • weak confluence, all critical pairs between rewrite rules have common reducts

• For non-terminating systems:
  • There are no overlaps (rules that give rise to critical pairs) in the first place, and...
  • The system is left-linear.
Not just for Confluence, UN & CON too:

\[
\begin{align*}
F(x,x) & \rightarrow A \\
F(C(x),x) & \rightarrow B \\
E & \rightarrow C(E)
\end{align*}
\]

F(E,E) has distinct normal forms A and B. So this is not UN, despite having no overlaps. Moreover, add the rules:

\[
\begin{align*}
G(A,x,y) & \rightarrow x \\
G(B,x,y) & \rightarrow y
\end{align*}
\]

Now, \( x = G(A,x,y) = G(F(E,E),x,y) = G(F(C(E),E),x,y) = G(B,x,y) = y \). The modified system still has no overlaps but is not CON.
What is going on?

• The system actually did have overlaps, but the standard definition of overlap does not acknowledge them:
  • If we allow for substitutions to replace variables with infinitary terms then the first two rules overlap, i.e. can be applied to the same term
  • And we had a finite term that was “semantical equal” to such an infinite term
• So if we move from LL & non-overlapping to non-\(\omega\)-overlapping then this counter-example goes away, but is that it?
  • We will not regain confluence, but UN and/or CON?
  • Open problem #79 since 1989.
Reducing the problem (i)

It suffices to look at CON:

• (On open terms) UN implies CON

• Suppose we had a non-$\omega$-overlapping TRS that was not UN. Then
  • There are distinct but equivalent normal forms $t$ and $u$.
  • We can get non-$\omega$-unifiable but equivalent ground normal forms $t'$ and $u'$ from $t$ and $u$, possibly via signature extension.
  • We add rules $G(t',x,y) \rightarrow x$, $G(u',x,y) \rightarrow y$, with new ternary symbol $G$.
  • The resulting system remains non-$\omega$-overlapping but it fails CON too.
Reducing the problem (ii)

• We can reduce the CON problem of a TRS to the CON problem of its constructor translation.

• The constructor translation of a TRS $T$ is:
  • A constructor TRS (first-order functional program) $T'$, with...
  • Back-and-forth translations between the terms of $T$ and $T'$, which preserves variables and...
  • preserves equations either way. (So this preserves and reflects CON.)
  • Aside: our translation also preserves and reflects SN (and WN).
Constructor Translation of a TRS

• Step 1: duplicate the signature; the new constructor TRS has a destructor $F_d$ and a constructor $F_c$ for every symbol $F$ of the original signature

• Step 2: for every old rule $F(p_1, \ldots, p_n) \rightarrow r$ we get a new rule:
  
  \[ F_d([p_1], \ldots, [p_n]) \rightarrow [r] \]

• Step 3: for every non-variable pattern $G(q_1, \ldots, q_k)$ (strict subterm of a left-hand side of an old rule) we get a new rule:
  
  \[ G_d([q_1], \ldots, [q_k]) \rightarrow G_c([q_1], \ldots, [q_k]) \]
Example

- Take Combinatory Logic: $K x y \rightarrow x, S x y z \rightarrow x z (y z)$.
- As a TRS this really is: $A(A(K, x), y) \rightarrow x, A(A(A(S, x), y), z) \rightarrow A(A(x, z), A(y, z))$.
- Constructor translation (slightly abbreviated) for this:
  
  $A_d(A_c(K, x), y) \rightarrow x$
  $A_d(K, x) \rightarrow A_c(K, x)$
  $A_d(A_c(A_c(S, x), y), z) \rightarrow A_d(A_d(x, z), A_d(y, z))$
  $A_d(A_c(S, x), y) \rightarrow A_c(A_c(S, x), y)$
  $A_d(S, x) \rightarrow A_c(S, x)$
What does the translation do to overlaps?

• The translation does not introduce overlaps, except between pattern rules.
• If a TRS is non-$\omega$-overlapping then its constructor-translation is “strongly almost non-$\omega$-overlapping”. This means: whenever two rules overlap then they are substitution instances of a common generalisation rule.
• For rules derived for patterns with root G we always have the generalisation $G_d(x_1, \ldots, x_n) \rightarrow G_c(x_1, \ldots, x_n)$
• This implies that all $\omega$-overlaps between rules are trivial (“almost non-$\omega$-overlapping”), but is even stronger than that.
Intermission: a tool for reasoning about terms

• Given a relation $R$ between terms, we write $\tilde{R}$ write for the relation on terms defined as:

$$t \tilde{R} u \equiv \exists F \in \Sigma, t_1, ..., t_n, u_1, ..., u_n. t = F(t_1, ..., t_n) \land u = F(u_1, ..., u_n) \land \forall i. t_i R u_i$$

• Similarly, $t\hat{R}u$ and $t\overline{R}u$ express the corresponding relations when the shared symbol $F$ is requested to be constructor (in $\Sigma_c$) or destructor (in $\Sigma_d$), respectively.

• A relation $R$ is called $\Sigma$-closed iff $\tilde{R} \subseteq R$. 
Observation: confluence vs. consistency

Why does confluence give us consistency?

• A system is confluent iff the joinability relation $\downarrow$ is transitive.

• The joinability relation can be defined like this:
  \[\downarrow \equiv \mu x. \text{id} \cup x \cup x^{-1} \cup \tilde{x} \cup \rightarrow_R \cdot x\]

• Thus: joinability is by construction reflexive, symmetric and $\Sigma$-closed, and contains rewrite steps. It is just short of transitivity from being a congruence.

• It is also by construction consistent.

Note: there are other relations that share these properties with $\downarrow$, so they could take its part in consistency proofs.
Computational invariants

• We typically prove confluence by showing that \( \downarrow \) is some kind of computational invariant. For this it needs to “survive” pattern matching. Relation-algebraically, it is this property:

• A relation \( R \) between terms is called constructor-compatible iff we have \( \hat{id} \cdot R \cdot \hat{id} \subseteq \hat{R} \)

• In long form: if two constructor-topped terms are related by \( R \) then they are topped by the same constructor and their direct subterms are pairwise related by \( R \)

• \( \downarrow \) is always constructor-compatible
Pattern Matching; Rule Application

• Let $p$ be a constructor term.
• Let $t = \sigma(p)$ and $u = \theta(p)$ be two substitution instances of $p$.
• If $t \mathcal{R} u$ and $\mathcal{R}$ is constructor-compatible then $\sigma$ and $\theta$ are pointwise related by $\mathcal{R}$. (on variables occurring in $p$)
• If in addition $\mathcal{R}$ is $\Sigma$-closed then it must survive parallel rule application with the same rule.
We need more though...

• How can we make sure though that parallel rule applications are with the same rule?

• We have this result: whenever two redexes t and u are related by $\equiv_c$, where $=_c$ is a constructor-compatible equivalence, then t and u are instances of two $\omega$-unifiable left-hand sides.

• Why? Informal reason: when we do $\omega$-unification of we perform some equational transformations. If the terms we unify are constructor terms then each transformational step is sound for any constructor-compatible equivalence.
Another invariant

• The semi-joinability relation can be defined like this:
  \[ \downarrow \equiv \mu x. id \cup x \cup x^{-1} \cup \bar{x} \cup \bar{x} \cdot x \cup \rightarrow \cdot x \]

• So, this relation \( \downarrow \) is reflexive, symmetric, \( \Sigma \)-closed, closed under prefixing with root-rewrite-steps, and it is closed under prefixing with itself on subterms of destructor-topped terms.

• Regardless of TRS, this relation is also constructor-compatible (and therefore consistent – when we view variables as constructors).

• So, this gives us a more relaxed invariant for consistency proofs than joinability. So, if \( \downarrow \) is transitive then we are home and dry.
One key difference to joinability

- Joinability is closed under prefixing with $\rightarrow_R$, semi-joinability is closed under prefixing with $\sqsubseteq$ - which is a symmetric relation.
- This gives extra flexibility when trying to construct a common “semi-reduct”.
Term-coalgebras

• $\Sigma$-coalgebras are sets whose elements (nodes) are term-like objects.
  • We may have additional structure, e.g. node labels.
  • The terms associated with nodes could be infinitary, and we may have the same term associated with more than one node.

• Term-coalgebras (for $\Sigma$) is the special case of sets of finite terms, closed under subterms.

• The $\tilde{R}$ notations carry over naturally to term-coalgebras (and indeed arbitrary $\Sigma$-coalgebras).
Transporting definitions

• We can view relations such as $\Downarrow$ as being defined (in the same way), for a particular coalgebra $A$.

• However, $\Downarrow_A$ is not just the restriction of $\Downarrow$ to $A \times A$, because $A$ is not required to include all terms – a redex may lose redex-status.

• In any case, $\Downarrow_A$ (on a term-coalgebra $A$) is a subrelation of $\Downarrow$ - because of monotonicity of the construction.

• Generally, if $t \Downarrow u$ holds then it is also the case that $t \Downarrow_A u$ for some finite term-coalgebra $A$. 
Constructing an equivalence

To prove that $\Downarrow_A$ is an equivalence for a finite term coalgebra $A$ we simply build an equivalence relation which:

- is constructor-compatible,
- is a subrelation of $\Downarrow_A$,
- is $\Sigma$-closed and contains $\Downarrow_A^*$ as a subrelation, and which
- includes “sufficiently many” redex contractions
How do we build it?

- As a union/find structure (with proof annotations).
- The node set of the structure is all of A.
- An edge from $a$ to $b$ requires that either $a \Downarrow b$ or $a \rightarrow b$ or $a \equiv_e b$ where $\equiv_e$ is the equivalence defined by the structure.
- We merge equivalence classes by adding an edge to a root of the structure that points to another class.
- We prioritise $\Downarrow$ edges over redex edges.
In the end

• We cannot add any more edges to merge equivalence classes...
• $e = \overline{\equiv}$ will have covered all of $\downarrow_A^*$ (because of our priorities)
• We can allow for no more than one redex-contraction per equivalence class of $\downarrow_A^*$, and we can ensure one is possible if there is one.
• That suffices for rewrite systems which have $\downarrow$ as their “consistency invariant”:
  whenever two parallel redexes are related by $\equiv_c$ and $=_c \subseteq \downarrow$ is any constructor-compatible equivalence, then their contracta are related by the $\Sigma$-closure of $=_c$. 
Consequences

• $\Downarrow_A$ is transitive (for finite A) for “well-behaved” TRSs

• $\Downarrow$ is transitive for well-behaved TRSs too:
  • If $t \Downarrow u$ and $u \Downarrow v$ then for some finite term-coalgebras A and B we have $t \Downarrow_A u$ and $u \Downarrow_B v$.
  • But $C = A \cup B$ is then a term-coalgebra too, we get $t \Downarrow_C u$ and $u \Downarrow_C v$ by monotonicity, $t \Downarrow_C v$ by transitivity of $\Downarrow_C$, and $t \Downarrow v$ by monotonicity.

• Thus, “well-behaved” TRSs are consistent.

• Strongly almost non-$\omega$-overlapping Constructor TRSs are “well-behaved”

• Therefore: non-$\omega$-overlapping TRSs have unique normal forms.
Future Work

Almost non-$\omega$-overlapping Constructor TRSs

Parallel steps could be with different rules, but we should still have our consistency invariant property.

Relaxing the condition on constructor-compatible equivalences:

We currently require that for all constructor-compatible sub-equivalences $S$ of $\downarrow$ that $CT(S)$ holds for the contracta whenever $\bar{S}$ holds between redexes.

But: one does not need “all”, one only needs “all that are sufficiently large”.