A Confluência como Consequência da Propriedade Z em Coq

Flávio L. C. de Moura[1] Leandro O. Rezende[2]

Departamento de Ciência da Computação, Universidade de Brasília, Brasília, Brasil

Abstract

A Teoria de Reescrita é um modelo computacional equivalente às máquinas de Turing, e seu sistema de reescrita de termos melhor conhecido é o cálculo-λ. A Confluência é uma propriedade importante e indecidível relacionada ao determinismo no processo computacional. Provas diretas de confluência em sistemas de reescrita de termos são, no geral, difíceis de serem feitas. Portanto, caracterizações alternativas da confluência podem ajudar a contornar essa dificuldade em contextos distintos. Esse é o caso da propriedade Z, que foi usada para provar confluência de diversos sistemas de reescrita, como o cálculo-λ com $\beta\eta$-redução, extensões do cálculo-λ com substituições explícitas, o cálculo-$\lambda\mu$, etc. Apresentamos nesse artigo uma prova construtiva de que a propriedade Z implica na confluência. As provas conhecidas desse fato geralmente dependem da lei do terceiro excluído, mas seu uso não é necessário, como mostramos em nossa prova, que contorna essa necessidade usando indução aninhada no passo indutivo da prova original. Ademais, nós formalizamos nossa prova, além de uma extensão da propriedade Z conhecida como propriedade Z composicional, no assistente de provas Coq.

Keywords: Sistema de reescrita de termos, confluência, assistentes de provas, Coq

1 Introdução

A confluência é uma propriedade importante e indecidível acerca do determinismo de um processo computacional. Podemos dizer, nesse contexto, que um programa é confluente se quaisquer duas maneiras de avaliá-lo resultam na mesma resposta. No caso particular de sistemas de reescrita de termos (em inglês, "Abstract Rewriting Systems", ou ARS), que são o foco deste artigo, a confluência pode ser expressada de maneira elegante através do uso de diagramas, como veremos na próxima seção.

As contribuições deste artigo são as seguintes:

- Apresentamos uma prova costrutiva e baseada em indução aninhada de que a propriedade Z implica em confluência;
- A prova de que a propriedade Z implica na confluência é formalizada no assistente de provas Coq, e sua apresentação é feita através da intercalação da explicação em português com o código correspondende em Coq. Assim, os comentários são feitos diretamente nos arquivos em Coq usando o estilo de comentário do coqdoc. Nós acreditamos que essa abordagem pode ser interessante para quem não seja familiarizado com o assistente de provas Coq, já que a intercalação de código com comentários em português dá uma boa ideia de como eles se relacionam.
- Nós formalizamos uma extensão da propriedade Z, conhecida como propriedade Z composicional, descrita em [6].

2 The Z property implies Confluence

An ARS, say (A, R), is defined as a pair composed of a set A and binary operation over this set $R : A \times A$. Let $a, b : A$, we write $a \; R \; b$ or $a \to_R b$ to denote that $(a, b) \in R$, and we say that a R-reduces to b in one step

[1] Email:flaviomoura@unb.br

[2] Email:l-ordo.ab.chao@hotmail.com

This paper is electronically published in
Electronic Notes in Theoretical Computer Science
URL: www.elsevier.nl/locate/entcs

. The arrow notation will be prefered because it is more convenient for expressing reductions, so the reflexive transitive closure of a relation R, written as $\twoheadrightarrow_R$, is defined by the following inference rules:

$$\frac{}{a \twoheadrightarrow_R a}\ (refl) \qquad\qquad \frac{a \to_R b \qquad b \twoheadrightarrow_R c}{a \twoheadrightarrow_R c}\ (rtrans)$$

where a, b and c are universally quantified variables as one makes explicit in the corresponding Coq definition:

```
Inductive refltrans {A:Type} (R: Rel A) : A → A → Prop :=
| refl: ∀ a, (refltrans R) a a
| rtrans: ∀ a b c, R a b → refltrans R b c → refltrans R a c.
```

The rules named (*refl*) and (*rtrans*) are called *constructors* in the Coq definition. The first constructor states the reflexivity axiom for $\twoheadrightarrow_R$, while *rtrans* extends the reflexive transitive closure of R, if one has at least a one-step reduction. As a first example, let's have a look at the proof of transitivity of $\twoheadrightarrow_R$:

Lemma 2.1 *Let $\to_R$ be a binary relation over a set A. If $t \twoheadrightarrow_R u$ and $u \twoheadrightarrow_R v$ then $t \twoheadrightarrow_R v$, $\forall t, u, v \in A$.*

Although its simplicity, it will help us to explain the way we will relate English annotations with the proof steps. The corresponding lemma in Coq, named *refltrans_composition*, is stated as follows:

```
Lemma refltrans_composition {A} (R: Rel A): ∀ t u v, refltrans R t u → refltrans R u v → refltrans R t v.
```

This work is not a Coq tutorial, but our idea is that it should also be readable for those unfamiliar with the Coq proof Assistant. In addition, this paper is built directly from a Coq proof script, which means that we are forced to present the ideas and the results in a more organized and systematic way that is not necessarily the more pedagogical one. Therefore, we decided to comment the proof steps giving the general idea of what they do. The uncommented proof of the lemma *refltrans_composition* is as follows:

```
Proof.
 intros t u v.
 intros H1 H2.
 induction H1.
 - assumption.
 - apply rtrans with b.
   + assumption.
   + apply IHrefltrans; assumption.
Qed.
```

Notice that proofs are written between the reserved words `Proof` and `Qed`, each proof command finishes with a dot, and proofs can be structured with bullets. We now present the commented proof of the lemma *refltrans_composition*, by writing the idea of the work done by each Coq command line. This will be the approach followed in this paper.

`Proof.`

`intros` *t u v*. Let t,u and v be elements of type A (or be elements of the set A).

`intros` *H1 H2*. Assume that $t \twoheadrightarrow_R u$ (name this assumption *H1*) and $u \twoheadrightarrow_R v$ (name this assumption *H2*).

`induction` *H1*. The proof proceeds by induction on the hypothesis *H1*, i.e. by induction on $t \twoheadrightarrow_R u$. The structure of the proof context determines the shape of the induction hypothesis, and this fact will be essential to understand the inductive proof of the next theorem. As shown in Figure 1, *H1* and *H2* are the only hypothesis (the other lines are just declaration of variables), therefore the induction hypothesis subsumes *H2*.

`- assumption`. The first case is when $t \twoheadrightarrow_R u$ is generated by the constructor *refl*, which is an axiom and hence we are done.

`- apply` *rtrans* `with` *b*. The second case, i.e. the recursive case is more interesting because $t \twoheadrightarrow_R u$ is now generated by *rtrans*. This means that there exists an element, say b, such that $t \to_R b$ and $b \twoheadrightarrow_R u$. Therefore, in order to prove that $t \twoheadrightarrow_R u$, we can apply the rule *rtrans* taking b as the intermediary term. The proof of

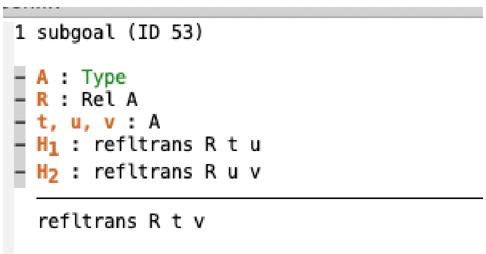


Figure 1. Transitivity of $\twoheadrightarrow_R$

the recursive case can be better visualized by the corresponding deduction tree:

$$\dfrac{\dfrac{\dfrac{}{\forall x\ y\ z, x \to_R y \to y \twoheadrightarrow_R z \to x \twoheadrightarrow_R z}\,rtrans}{t \to_R b \to b \twoheadrightarrow_R u \to t \twoheadrightarrow_R u}\,(\forall_e) \quad \dfrac{}{t \to_R b}\,\mathrm{H}}{b \twoheadrightarrow_R u \to t \twoheadrightarrow_R u}\,\mathrm{MP} \quad \dfrac{\dfrac{}{u \twoheadrightarrow_R v \to b \twoheadrightarrow_R u}\,IH \quad \dfrac{}{u \twoheadrightarrow_R v}\,\mathrm{H2}}{b \twoheadrightarrow_R u}\,\mathrm{MP}}{t \twoheadrightarrow_R u}\,\mathrm{MP}$$

Each branch of the above tree corresponds to a new goal in the Coq proof. Therefore, we have two subcases (or subgoals) to prove:

+ `assumption`. In this subgoal we need to prove that $t \to_R b$, which we have as hypothesis.

+ `apply` *IHrefltrans*; `assumption`. In the second subgoal, we need to prove that $b \twoheadrightarrow_R u$. To do so, we apply the induction hypothesis *IHrefltrans*: $u \twoheadrightarrow_R v \to b \twoheadrightarrow_R u$, where $u \twoheadrightarrow_R v$ is the hypothesis *H2*. `Qed`.

This example is interesting because it shows how Coq works, how each command line (also known as tactics or tacticals depending on its structure) corresponds, in general, to several steps of natural deduction rules.

The reflexive transitive closure of a relation is used to define the notion of confluence: no matter how the reduction is done, the result will always be the same. In other words, every divergence is joinable as stated by the following diagram:

$$\begin{tikzcd} & a \arrow[dl, two heads] \arrow[dr, two heads] & \\ b \arrow[dr, dotted, two heads] & & c \arrow[dl, dotted, two heads] \\ & d & \end{tikzcd}$$

Formally, this means that if an expression a can be reduced in two different ways to the expressions b and c, then there exists an expression d such that both b and c reduce to d. The existential quantification is expressed by the dotted lines in the diagram. This notion is defined in the Coq system as follows:

`Definition` *Confl* $\{A$:`Type`$\}$ (R: *Rel* A) := $\forall$ a b c, (*refltrans* R) a b $\to$ (*refltrans* R) a c $\to$ ($\exists$ d, (*refltrans* R) b d $\land$ (*refltrans* R) c d).

In [9], V. van Oostrom gives a suficient condition for an ARS to be confluent, known as the *Z Property*:

Definition 2.2 Let $(A, \to_R)$ be an ARS. Then $(A, \to_R)$ has the Z property, if there exists a map $f : A \to A$ such that the following diagram holds:

$$\begin{tikzcd} a \arrow[r, "R"'] & b \arrow[dl, dotted, two heads, "R"] \\ f(a) \arrow[r, dotted, two heads, "R"'] & f(b) \end{tikzcd}$$

The corresponding Coq definition is given as:

`Definition` *Z_prop* $\{A$:`Type`$\}$ (R: *Rel* A) := $\exists$ f:$A \to A$, $\forall$ a b, R a b $\to$ ((*refltrans* R) b (f a) $\land$ (*refltrans* R) (f a) (f b)).

Alternatively, when f satisfies the Z property, one says that f is Z:

`Definition` *f_is_Z* {*A*:Type} (*R*: *Rel A*) (*f*: *A* → *A*) := ∀ *a b*, *R a b* → ((*refltrans R*) *b* (*f a*) ∧ (*refltrans R*) (*f a*) (*f b*)).

The first contribution of this work is a constructive proof of the fact that the Z property implies confluence. Our proof uses nested induction, and hence it differs from the one in [5] (that follows [9]) in the sense that it does not rely on the law of the excluded middle. As a result, we have an elegant inductive proof of the fact that if a binary relation has the Z property then it is confluent. In addition, we formalized this proof in the Coq proof assistant. In [4], B. Felgenhauer et.al. formalized the Z property in Isabelle/HOL. In what follows, we present the theorem and its proof interleaving Coq code and the corresponding comments.

`Theorem` *Z_prop_implies_Confl* {*A*:Type}: ∀ *R*: *Rel A*, *Z_prop R* → *Confl R*.
`Proof.`

`intros` *R HZ_prop*. Let R be a relation over A that satisfies the Z property, which will be denoted by *HZ_prop* for future reference.

`unfold` *Z_prop*, *Confl* `in` *. Unfolding both definitions of *Z_prop* and *Confl*, we get the following proof context:

```
1 subgoal (ID 90)

- A : Type
- R : Rel A
- HZ_prop : ∃ f : A → A,
             ∀ a b : A, R a b → refltrans R b (f a) ∧ refltrans R (f a) (f b)
______________________________________________________________________
  ∀ a b c : A,
  refltrans R a b →
  refltrans R a c → ∃ d : A, refltrans R b d ∧ refltrans R c d
```

`intros` *a b c Hrefl1 Hrefl2*. Let a, b and c be elements of the set A, *Hrefl1* the hypothesis that $a \twoheadrightarrow_R b$, and *Hrefl2* the hypothesis that $a \twoheadrightarrow_R c$. We need to prove that there exists d such that $b \twoheadrightarrow_R d$ and $c \twoheadrightarrow_R d$

`destruct` *HZ_prop* `as` [*g HZ_prop*]. We know from the hypothesis *HZ_prop* that there exists a mapping f that is Z. Let's call g this mapping, and we get following proof context:

```
1 subgoal (ID 103)

- A : Type
- R : Rel A
- g : A → A
- HZ_prop : ∀ a b : A, R a b → refltrans R b (g a) ∧ refltrans R (g a) (g b)
- a, b, c : A
- Hrefl1 : refltrans R a b
- Hrefl2 : refltrans R a c
______________________________________________________________________
  ∃ d : A, refltrans R b d ∧ refltrans R c d
```

The proof proceeds by nested induction, firstly on the length of the reduction from a to b, and then on the length of the reduction from a to c.

`generalize dependent` *c*. Before the first induction, i.e. induction on *Hrefl1*, the element c needs to be generalized so that it can be afterwards instantiated with any reduct of a.

`induction` *Hrefl1*. The induction on *Hrefl1* corresponds to induction on the reflexive transitive closure of the relation R, and since *refltrans* has two rules, the goal splits in two subgoals, one for each possible way of constructing $a \twoheadrightarrow_R b$.

- `intros` *c Hrefl2*. In the first case, we have that $b = a$ since we are in the reflexive case. This means that we have to prove that there exists d, such that $a \twoheadrightarrow_R d$ and $c \twoheadrightarrow_R d$.

  ∃ *c*; `split`. Taking d as c, the proof is simplified to $a \twoheadrightarrow_R c$ and $c \twoheadrightarrow_R c$.

  + `assumption`. The first component is exactly the hypothesis *Hrefl2* and,

  + `apply` *refl*. $c \twoheadrightarrow_R c$ corresponds to an application of the *refl* axiom.

The interesting part of the proof is then given by the inductive case, i.e. when $a \twoheadrightarrow_R b$ is generated by the rule (*rtrans*). In this case, the reduction from a to b is done in at least one step, therefore there must exists an element a' such that the following diagram holds.

$$
\begin{tikzcd}
 & & a \arrow[dl] \arrow[dr, two heads] & \\
 & a' \arrow[dl, two heads] & & c \arrow[ddll, dotted, two heads] \\
b \arrow[dr, dotted, two heads] & & & \\
 & d & &
\end{tikzcd}
$$

The induction hypothesis states that every divergence from a', that reduces to b from one side, converges: *IHHrefl1* : $\forall c_0 : A, a' \twoheadrightarrow_R c_0 \to (\exists d : A, b \twoheadrightarrow_R d \wedge c_0 \twoheadrightarrow_R d)$. The idea is to apply induction on the hypothesis *Hrefl2*, but the current proof context has the hypothesis H: $a \to_R a'$ (a reduces to a' in one step), and hence it is the sole hypothesis depending on a in the current proof context. Therefore, suppose that $a \twoheadrightarrow_R c$ is built as $a \to_R a'' \twoheadrightarrow_R c$ in the case of the constructor *rtrans*. The induction step in this case will assume that a'' reduces in one step to a', which is not true in general. Note that all hypothesis, that do not have a as parameter, do not contribute to the shape of the induction hypothesis. In order to circumvent this problem, we need to remove the hypothesis H, and replace it by another relevant information derived from the Z property as shown in what follows.

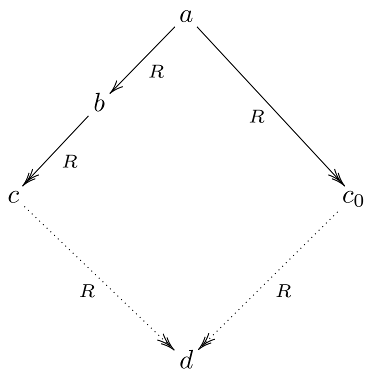
`- intros` *c0 Hrefl2*. Let c_0 be a reduct of a, and *Hrefl2* be the hypothesis $a \twoheadrightarrow_R c_0$. So the reduction $a \twoheadrightarrow_R c$ in the above diagram is now $a \twoheadrightarrow_R c_0$ due to a renaming of variables automatically done by the Coq system. In addition, the reduction $a \to_R a' \twoheadrightarrow_R b$ is now $a \to_R b \twoheadrightarrow_R c$, as shown below:

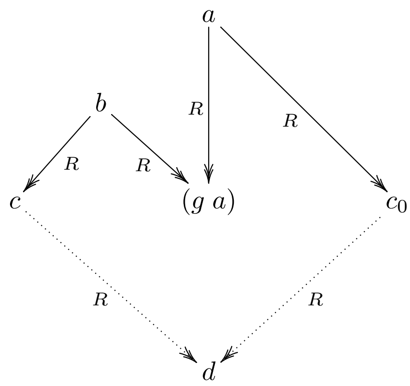
```
1 subgoal (ID 130)

- A : Type
- R : Rel A
- g : A → A
- HZ_prop : ∀ a b : A, R a b → refltrans R b (g a) ∧ refltrans R (g a) (g b)
- a, b, c : A
- H : R a b
- Hrefl1 : refltrans R b c
- IHHrefl1 : ∀ c0 : A,
             refltrans R b c0 → ∃ d : A, refltrans R c d ∧ refltrans R c0 d
- c0 : A
- Hrefl2 : refltrans R a c0
______________________________________________________________________
  ∃ d : A, refltrans R c d ∧ refltrans R c0 d
```

Before applying induction to *Hrefl2*: $a \twoheadrightarrow_R c_0$, we will be replace the hypothesis H: $a \to_R b$ by two other properties that are proved from the Z property: $b \twoheadrightarrow_R (g\ a)$ and $a \twoheadrightarrow_R (g\ a)$.

`assert` (*Hbga*: *refltrans R b* $(g\ a)$).
{ `apply` *HZ_prop*; `assumption`. } We call *Hbga* the reduction $b \twoheadrightarrow_R (g\ a)$ that is directly obtained from the Z property.
`assert` (*Haga*: *refltrans R a* $(g\ a)$).
{ `apply` *rtrans* `with` b; `assumption`. } Call *Haga* the reduction $a \twoheadrightarrow_R (g\ a)$, and prove it using the transitivity of $\twoheadrightarrow_R$, since $a \to_R b$ and $b \twoheadrightarrow_R (g\ a)$.

Diagrammatically, we change from the situation on the left to the one on the right:

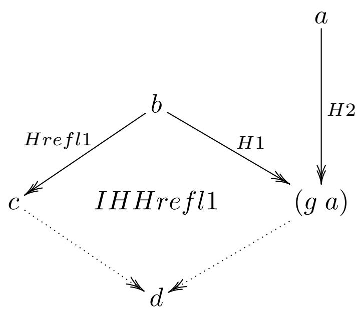




`clear` H; `generalize dependent` b. At this point we can remove the hypothesis H from the context, and generalize b.

`induction` *Hrefl2*. Now we are ready to start the induction on the reduction $a \twoheadrightarrow_R c_0$, and we have two subgoals.

$+$ `intros` *b Hrefl1 IHHrefl1 Hbga*. The first subgoal corresponds to the reflexive case that is closed by the induction hypothesis *IHHrefl1*:



`assert` (*IHHrefl1_ga* := *IHHrefl1* $(g\ a)$); `apply` *IHHrefl1_ga* `in` *Hbga*. In order to apply *IHHrefl1*, we instantiate c_0 with $(g\ a)$.

`destruct` *Hbga*. Therefore, there exists an element, say x, such that both $c \twoheadrightarrow_R x$ and $(g\ a) \twoheadrightarrow_R x$.

$\exists\ x$; `split`. We then take x to show that $c \twoheadrightarrow_R x$ and $a \twoheadrightarrow_R x$.

$\times$ `apply` H. Note that $c \twoheadrightarrow_R x$ is already an hypothesis, and we are done.

$\times$ `apply` *refltrans_composition* `with` $(g\ a)$; [`assumption` | `apply` H]. The proof of $a \twoheadrightarrow_R x$ is done by the transitivity of $\twoheadrightarrow_R$ taking $(g\ a)$ as the intermediary step.

$+$ `intros` *b0 Hrefl1 IHHrefl1 Hb0ga*. The second subgoal corresponds to the case in which $a \twoheadrightarrow_R c_0$ is generated by the rule (*rtrans*). Therefore, there exists a term b such that $a \to_R b$ and $b \twoheadrightarrow_R c_0$. The corresponding proof context after introducing the universally quantified variable *b0*, the hypothesis *Hrefl1* and the induction hypothesis *IHHrefl1* generated by the first outer induction and the fact that $b0 \twoheadrightarrow_R (g\ a)$ is given by:

```
1 subgoal (ID 188)

- A : Type
- R : Rel A
- g : A → A
- HZ_prop : ∀ a b : A, R a b → refltrans R b (g a) ∧ refltrans R (g a) (g b)
- c, a : A
- H2 : refltrans R a (g a)
- b : A
- Hrefl1 : refltrans R b c
- IHHrefl1 : ∀ c0 : A,
             refltrans R b c0 → ∃ d : A, refltrans R c d ∧ refltrans R c0 d
- x : A
- H : refltrans R c x ∧ refltrans R (g a) x
- IHHrefl1_ga : refltrans R b (g a) →
                ∃ d : A, refltrans R c d ∧ refltrans R (g a) d
  ______________________________________________________________________
  ∃ d : A, refltrans R c d ∧ refltrans R a d
```

`apply` *IHHrefl2* `with` *b0.* The second goal, i.e. the inductive case can be proved by the second induction hypothesis *IHHrefl2*, and each of the 4 conditions generated by this hypothesis is solved as follows:

$\times$ `apply` *refltrans_composition* `with` $(g\ a)$; `apply` *HZ_prop*; `assumption`. 1. $b \twoheadrightarrow_R (g\ b)$: This is proved by the transitivity of the reflexive transitive closure of R using the hypothesis (H: $a \to_R b$) and *HZ_prop*: $\forall a\ b : a \to_R b \to (b \twoheadrightarrow_R (g\ a) \wedge (g\ a) \twoheadrightarrow_R (g\ b))$.

$\times$ `assumption`. 2. $b0 \twoheadrightarrow_R c$: This is exactly the hypothesis *Hrefl1.*

$\times$ `assumption`. 3. $\forall c0 : b0 \twoheadrightarrow_R c0 \to (\exists d : c \twoheadrightarrow_R d \wedge c0 \twoheadrightarrow_R d)$: This is exactly the induction hypothesis *IHHrefl1.*

$\times$ `apply` *refltrans_composition* `with` $(g\ a)$; [`assumption` | `apply` *HZ_prop*; `assumption`]. 4. $b0 \twoheadrightarrow_R (g\ b)$: This is proved by the transitivity of the reflexive transitive closure of R using the hypothesis (*H'*: $b0 \twoheadrightarrow_R (g\ a)$ and the fact that $(g\ a) \twoheadrightarrow_R (g\ b)$ that is obtained from the fact that R satisfies the Z property (hypothesis *HZ_prop*).

`Qed.`

An alternative proof that Z implies confluence is possible via the notion of semiconfluence, which is equivalent to confluence, as done in [4]. Our proof is also constructive, but we will not explain it here due to lack of space; any interested reader can find it in the Coq file in our GitHub repository.

`Definition` *SemiConfl* $\{A$:Type$\}$ (R: *Rel* A) := $\forall\ a\ b\ c$, $R\ a\ b \to$ (*refltrans* R) $a\ c \to (\exists\ d$, (*refltrans* R) $b\ d$ $\wedge$ (*refltrans* R) $c\ d)$.

`Theorem` *Z_prop_implies_SemiConfl* $\{A$:Type$\}$: $\forall$ R: *Rel* A, *Z_prop* $R \to$ *SemiConfl* R.

`Theorem` *Semi_equiv_Confl* $\{A$: `Type`$\}$: $\forall$ R: *Rel* A, *Confl* $R \leftrightarrow$ *SemiConfl* R.

`Corollary` *Zprop_implies_Confl_via_SemiConfl* $\{A$:Type$\}$: $\forall$ R: *Rel* A, *Z_prop* $R \to$ *Confl* R.

```
Proof.
  intros R HZ_prop.
  apply Semi_equiv_Confl.
  generalize dependent HZ_prop.
  apply Z_prop_implies_SemiConfl.
Qed.
```

3 An extension of the Z property: Compositional Z

In this section we present a formalization of an extension of the Z property with compositional functions, known as *Compositional Z*, as presented in [6]. The compositional Z is an interesting property because it allows a kind of modular approach to the Z property in such a way that the reduction relation can be split into two parts. More precisely, given an ARS $(A, \to_R)$, one must be able to decompose the relation $\to_R$ into two parts, say $\to_1$ and $\to_2$ such that $\to_R = \to_1 \cup \to_2$. This kind of decomposition can be done in several interesting situations such as the λ-calculus with $\beta\eta$-reduction[3], extensions of the λ-calculus with explicit substitutions[1], the $\lambda\mu$-calculus[7], etc. But before presenting the complete definition of the Compositional Z, we need to define the *weak Z property*:

$$\begin{tikzcd} a \arrow[r, "R"'] & b \arrow[dl, dotted, two heads, "x"] \\ f(a) \arrow[r, dotted, two heads, "x"'] & f(b) \end{tikzcd}$$

Figure 2. The weak Z property

Definition 3.1 Let $(A, \to_R)$ be an ARS and $\to_x$ a relation on A. A mapping f satisfies the *weak Z property* for $\to_R$ by $\to_x$ if $a \to_R b$ implies $b \twoheadrightarrow_x f(a)$ and $f(a) \twoheadrightarrow_x f(b)$ (cf. Figure 2). Therefore, a mapping f satisfies the Z property for $\to_R$ if it satisfies the weak Z property by itself.

When f satisfies the weak Z property, we also say that f is weakly Z, and the corresponding definition in Coq is given as follows:

`Definition` *f_is_weak_Z* $\{A\}$ (*R R'*: *Rel A*) (*f*: $A \to A$) := $\forall$ *a b*, *R a b* $\to$ ((*refltrans R'*) *b* (*f a*) $\wedge$ (*refltrans R'*) (*f a*) (*f b*)).

The compositional Z is an extension of the Z property for compositional functions, where composition is defined as usual:

`Definition` *comp* $\{A\}$ (*f1 f2*: $A \to A$) := `fun` *x*:*A* $\Rightarrow$ *f1* (*f2 x*).
`Notation` "f1 # f2" := (*comp f1 f2*) (`at level` 40).

and the disjoint union is inductively defined as:

`Inductive` *union* $\{A\}$ (*red1 red2*: *Rel A*) : *Rel A* :=
| *union_left*: $\forall$ *a b*, *red1 a b* $\to$ *union red1 red2 a b*
| *union_right*: $\forall$ *a b*, *red2 a b* $\to$ *union red1 red2 a b*.
`Notation` "R1 !_! R2" := (*union R1 R2*) (`at level` 40).

We are now ready to present the definition of the compositional Z:

Theorem 3.2 *[6] Let $(A, \to_R)$ be an ARS such that $\to_R = \to_1 \cup \to_2$. If there exists mappings $f_1, f_2 : A \to A$ such that*

(i) *f_1 is Z for $\to_1$*

(ii) *$a \to_1 b$ implies $f_2(a) \twoheadrightarrow f_2(b)$*

(iii) *$a \twoheadrightarrow f_2(a)$ holds for any $a \in Im(f_1)$*

(iv) *$f_2 \circ f_1$ is weakly Z for $\to_2$ by $\to_R$*

then $f_2 \circ f_1$ is Z for $(A, \to_R)$, and hence $(A, \to_R)$ is confluent.

We define the predicate *Z_comp* that corresponds to the premises of Theorem 3.2, i.e. to the conjunction of items (i), (ii), (iii) and (iv) in addition to the fact that $\to_R = \to_1 \cup \to_2$, where $\to_1$ (resp. $\to_2$) is written as *R1* (resp. *R2*):

`Definition` *Z_comp* $\{$*A*:`Type`$\}$ (*R* :*Rel A*) := $\exists$ (*R1 R2*: *Rel A*) (*f1 f2*: $A \to A$), *R* = (*R1* !_! *R2*) $\wedge$ *f_is_Z R1 f1* $\wedge$ ($\forall$ *a b*, *R1 a b* $\to$ (*refltrans R*) (*f2 a*) (*f2 b*)) $\wedge$ ($\forall$ *a b*, *b* = *f1 a* $\to$ (*refltrans R*) *b* (*f2 b*)) $\wedge$ (*f_is_weak_Z R2 R* (*f2* # *f1*)).

As stated by Theorem 3.2, the compositional Z gives a sufficient condition for compositional functions to be Z. In other words, compositional Z implies Z, which is justified by the diagrams of Figure 3.

$$\begin{tikzcd} a \arrow[rr, "1"] & & b \arrow[dll, dotted, two heads, "1"'] \\ f_1(a) \arrow[rr, dotted, two heads, "1"] \arrow[d, dotted] & & f_1(b) \\ f_2(f_1(a)) \arrow[rr, dotted, two heads] & & f_2(f_1(b)) \end{tikzcd} \qquad \begin{tikzcd} a \arrow[rr, "2"] & & b \arrow[dll, dotted, two heads] \\ f_2(f_1(a)) \arrow[rr, dotted, two heads] & & f_2(f_1(b)) \end{tikzcd}$$

Figure 3. Compositional Z implies Z

In what follows, we present our commented Coq proof of this fact:

`Theorem` *Z_comp_implies_Z_prop* $\{A$:`Type`$\}$: $\forall$ $(R$:*Rel* $A)$, *Z_comp* $R \to$ *Z_prop* R.
`Proof`.
`intros` R H. Let R be a relation over A, and H the hypothesis that R satisfies the compositional Z.

`unfold` *Z_prop*. `unfold` *Z_comp* `in` H. `destruct` H `as` [*R1* [*R2* [*f1* [*f2* [*Hunion* [*H1* [*H2* [*H3* *H4*]]]]]]]]. Now unfold the definitions of *Z_prop* and *Z_comp* as presented before, and name the hypothesis of the compositional Z as in Theorem 3.2. We need to prove that there exists a map, say f, that is Z as shown by the current proof context:

```
1 subgoal (ID 167)

A : Type
R, R1, R2 : Rel A
f1, f2 : A → A
Hunion : R = R1 !_! R2
H1 : f_is_Z R1 f1
H2 : ∀ a b : A, R1 a b → refltrans R (f2 a) (f2 b)
H3 : ∀ a b : A, b = f1 a → refltrans R b (f2 b)
H4 : f_is_weak_Z R2 R (f2 # f1)
______________________________________
∃ f : A → A,
  ∀ a b : A, R a b → refltrans R b (f a) ∧ refltrans R (f a) (f b)
```

$\exists$ (*f2* # *f1*). We will prove that the composition $f_2 \circ f_1$ is Z.

`intros` a b *HR*. Let a and b be elements of A, and suppose that a R-reduces to b in one step, i.e. that $a \to_R b$ and call *HR* this hypothesis.

`inversion` *Hunion*; `subst`. `clear` H. `inversion` *HR*; `subst`. Since R is the union of *R1* and *R2*, one has that a reduces to b in one step via either *R1* or *R2*. Therefore, there are two cases to consider:

- `split`. Firstly, suppose that a *R1*-reduces in one step to b, i.e. $a \to_{R1} b$.

$+$ `apply` *refltrans_composition* `with` (*f1* a). In order to prove that $b \twoheadrightarrow_R (f_2(f_1\ a))$, we first need to show that $b \twoheadrightarrow_{R1} (f_1\ a)$, and then that $(f_1\ a) \twoheadrightarrow_R (f_2(f_1\ a))$ as shown in Figure 3.

$\times$ `apply` *H1* `in` H. `destruct` H. `apply` *refltrans_union*; `assumption`. The proof of $b \twoheadrightarrow_{R1} (f_1\ a)$ is done from the fact that f_1 is Z for *R1*.

$\times$ `apply` *H3* `with` a; `reflexivity`. The proof that $(f_1\ a) \twoheadrightarrow_R (f_2(f_1\ a))$ is a direct consequence of the hypothesis *H3*.

$+$ `apply` *H1* `in` H. `destruct` H. `clear` H *HR*. `unfold` *comp*. The proof that $(f_2(f_1\ a))$ R-reduces to $(f_2(f_1\ b))$ is more tricky. Initially, note that, since $a \to_{R1} b$ then we get that $(f_1\ a) \twoheadrightarrow_{R1} (f_1\ b)$ by the Z property.

`induction` *H0*. Now, the goal can be obtained from *H2* as long as $(f_1\ a) \to_{R1} (f_1\ b)$, but from the hypothesis *H0* we have that $(f_1\ a) \twoheadrightarrow_{R1} (f_1\ b)$. Therefore, we proceed by induction on *H0*.

$\times$ `apply` *refl*. The reflexive case is trivial because a and b are equal.

$\times$ `apply` *refltrans_composition* `with` (*f2* *b0*). In the transitive case, we have that $(f_1\ a)$ *R1*-reduces to $(f_1\ b)$ in at least one step. The current proof context is as follows, up to renaming of variables:

```
1 subgoal (ID 314)

- A : Type
- R₁, R₂ : Rel A
- f₁, f₂ : A → A
- H₁ : f_is_Z R₁ f₁
- H₄ : f_is_weak_Z R₂ (R₁ !_! R₂) (f₂ # f₁)
- H₃ : ∀ a b : A, b = f₁ a → refltrans (R₁ !_! R₂) b (f₂ b)
- H₂ : ∀ a b : A, R₁ a b → refltrans (R₁ !_! R₂) (f₂ a) (f₂ b)
- a, b, a₀, b₀, c : A
- H : R₁ a₀ b₀
- H₀ : refltrans R₁ b₀ c
- IHrefltrans : refltrans (R₁ !_! R₂) (f₂ b₀) (f₂ c)
  ______________________________________
  refltrans (R₁ !_! R₂) (f₂ a₀) (f₂ c)
```

Therefore, there exists some element $b0$ such that $a0 \to_{R1} b0$ and $b0 \twoheadrightarrow_{R1} c$ and we need to prove that $(f_2\ a0) \twoheadrightarrow_{R1 \cup R2} (f_2\ c)$. This can be done in two steps using the transitivity of *refltrans* taking $(f_2\ b0)$ as the intermediary term.

** `apply` *H2*; `assumption`. The first subgoal is then $(f_2\ a0) \twoheadrightarrow_{(R1 \cup R2)} (f_2\ b0)$ that is proved by hypothesis *H2*.

** `assumption`. And the second subgoal $(f_2\ b0) \twoheadrightarrow_{(R1 \cup R2)} (f_2\ c)$ is proved by the induction hypothesis.

- `apply` *H4*; `assumption`. Finally, when a *R2*-reduces in one step to b one concludes the proof using the assumption that $(f_2 \circ f_1)$ is weak Z. `Qed`.

Now we can use the proofs of the theorems *Z_comp_implies_Z_prop* and *Z_prop_implies_Confl* to conclude that compositional Z is a sufficient condition for confluence.

`Corollary` *Z_comp_is_Confl* $\{A\}$: $\forall$ (*R*: *Rel A*), *Z_comp R* $\to$ *Confl R*.
`Proof`.
  `intros` *R H*.
  `apply` *Z_comp_implies_Z_prop* `in` *H*.
  `apply` *Z_prop_implies_Confl*; `assumption`.
`Qed`.

Rewriting Systems with equations is another interesting and non-trivial topic [10,8]. The confluence of rewriting systems with an equivalence relation can also be proved by a variant of the compositional Z, known as Z property modulo [2].

Corollary 3.3 *[6] Let $(A, \to_R)$ be an ARS such that $\to_R = \to_1 \cup \to_2$. If there exist mappings $f_1, f_2 : A \to A$ such that*

(i) $a \to_1 b$ *implies* $f_1(a) = f_1(b)$
(ii) $a \twoheadrightarrow_1 f_1(a), \forall a$
(iii) $a \twoheadrightarrow_R f_2(a)$ *holds for any* $a \in Im(f_1)$
(iv) $f_2 \circ f_1$ *is weakly Z for* $\to_2$ *by* $\to_R$

then $f_2 \circ f_1$ is Z for $(A, \to_R)$, and hence $(A, \to_R)$ is confluent.

We define the predicate *Z_comp_eq* corresponding to the hypothesis of Corollary 3.3, and then we prove directly that if *Z_comp_eq* holds for a relation R then *Zprop* R also holds. This approach differs from [6] that proves Corollary 3.3 directly from Theorem 3.2

`Definition` *Z_comp_eq* $\{A$:`Type`$\}$ (*R* :*Rel A*) := $\exists$ (*R1 R2*: *Rel A*) (*f1 f2*: $A \to A$), $R = (R1\ !\_!\ R2) \wedge (\forall\ a\ b,\ R1\ a\ b \to (f1\ a) = (f1\ b)) \wedge (\forall\ a,\ (refltrans\ R1)\ a\ (f1\ a)) \wedge (\forall\ b\ a,\ a = f1\ b \to (refltrans\ R)\ a\ (f2\ a)) \wedge (f\_is\_weak\_Z\ R2\ R\ (f2\ \#\ f1))$.

`Lemma` *Z_comp_eq_implies_Z_prop* $\{A$:`Type`$\}$: $\forall$ (*R* : *Rel A*), *Z_comp_eq R* $\to$ *Z_prop R*.
`Proof`.
  `intros` *R Heq*. `unfold` *Z_comp_eq* `in` *Heq*. Let R be a relation and suppose that R satisfies the predicate *Z_comp_eq*.

`destruct` *Heq* `as` [*R1* [*R2* [*f1* [*f2* [*Hunion* [*H1* [*H2* [*H3* *H4*]]]]]]]]. Call *Hi* the ith hypothesis as in 3.3.

`unfold` *Z_prop*. $\exists$ (*f2* # *f1*). From the definition of the predicate *Z_prop*, we need to find a map, say f that is Z. Let $(f_2 \circ f_1)$ be such map.

`intros` *a b Hab*. In order to prove that $(f_2 \circ f_1)$ is Z, let a and b be arbitrary elements of type A, and *Hab* be the hypothesis that $a \to_R b$.

`inversion` *Hunion*; `subst`; `clear` *H*. `inversion` *Hab*; `subst`; `clear` *Hab*. Since a R-reduces in one step to b and R is the union of the relations *R1* and *R2* then we consider two cases:

- `unfold` *comp*; `split`. The first case is when $a \to_{R1} b$. This is equivalent to say that $f_2 \circ f_1$ is weak Z for $R1$ by $R1 \cup R2$.

  + `apply` *refltrans_composition* `with` (*f1 b*). Therefore, we first prove that $b \twoheadrightarrow_{(R1\cup R2)} (f_2(f_1\ a))$, which can be reduced to $b \twoheadrightarrow_{(R1\cup R2)} (f_1\ b)$ and $(f_1\ b) \twoheadrightarrow_{(R1\cup R2)} (f_2(f_1\ a))$ by the transitivity of *refltrans*.

    × `apply` *refltrans_union*. `apply` *H2*. From hypothesis *H2*, we know that $a \twoheadrightarrow_{R1} (f_1\ a)$ for all a, and hence $a \twoheadrightarrow_{(R1\cup R2)} (f_1\ a)$ and we conclude.

    × `apply` *H1* `in` *H*. `rewrite` *H*. `apply` *H3* `with` *b*; `reflexivity`. The proof that $(f_1\ b) \twoheadrightarrow_{(R1\cup R2)} (f_2(f_1\ a))$ is exactly the hypothesis *H3*.

  + `apply` *H1* `in` *H*. `rewrite` *H*. `apply` *refl*. The proof that $(f_2(f_1\ a)) \twoheadrightarrow_{(R1\cup R2)} (f_2(f_1\ b))$ is done using the reflexivity of *refltrans* because $(f_2(f_1\ a)) = (f_2(f_1\ b))$ by hypothesis *H1*.

- `apply` *H4*; `assumption`. When $a \to_{R2} b$ then we are done by hypothesis *H4*.

`Qed`.

4 Conclusão

Nesse trabalho, apresentamos uma prova construtiva de que a propriedade Z implica na confluência, uma propriedade interessante para sistemas de reescrita de termos. Além disso, nós provamos esse resultado formalmente usando o assistente de provas Coq. Os arquivos contendo esses resultados estão disponíveis no nosso repositório do GitHub: https://github.com/flaviodemoura/Zproperty.

A propriedade Z foi apresentada inicialmente por V. van Oostrom como condição suficiente para um ARS ser confluente [4], e desde então foi usada para provar confluência em diversos contextos, como o cálculo-λ com $\beta\eta$-redução, extensões do cálculo-λ com substituições explícitas e o cálculo $\lambda\mu$. As provas em Coq dos resultados principais foram comentadas linha a linha, servindo tanto como uma apresentação informal das provas quanto como seu complemento formal. Ademais, nós formalizamos uma extensão da propriedade Z, conhecida como propriedade Z composicional, descrita em [6].

Futuramente, essa formalização será usada para provar a confluência de um cálculo com substituições explícitas baseado no cálculo-λ_{ex} (cf. [5]). Por fim, esperamos que nossa formalização possa ser usada como um framework para prova de confluência em outros sistemas de reescrita.

References

[1] M. Abadi, L. Cardelli, P.-L. Curien, and J.-J. Lévy. Explicit Substitutions. *Journal of Functional Programming*, 1(4):375–416, 1991.

[2] B. Accattoli and D. Kesner. The permutative lambda calculus. In Nikolaj Bjørner and Andrei Voronkov, editors, *LPAR*, volume 7180 of *Lecture Notes in Computer Science*, pages 23–36. Springer, 2012.

[3] H. P. Barendregt. *The Lambda Calculus : Its Syntax and Semantics (revised edition)*. North Holland, 1984.

[4] B. Felgenhauer, J. Nagele, V. van Oostrom, and C. Sternagel. The Z property. *Archive of Formal Proofs*, 2016.

[5] D. Kesner. A Theory of Explicit Substitutions with Safe and Full Composition. *Logical Methods in Computer Science*, 5(3:1):1–29, 2009.

[6] Koji Nakazawa and Ken-etsu Fujita. Compositional Z: confluence proofs for permutative conversion. *Studia Logica*, 104(6):1205–1224, 2016.

[7] Michel Parigot. Lambda-mu-calculus: An algorithmic interpretation of classical natural deduction. In Andrei Voronkov, editor, *Logic Programming and Automated Reasoning,International Conference LPAR'92, St. Petersburg, Russia, July 15-20, 1992, Proceedings*, volume 624 of *Lecture Notes in Computer Science*, pages 190–201. Springer, 1992.

[8] Terese. *Term Rewriting Systems*, volume 55 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2003.

[9] Vincent van Oostrom. Z - draft: For your mind only. 2007.

[10] Franz Winkler. *Equational Theorem Proving and Rewrite Rule Systems*, pages 26–39. Informatik-Fachberichte. Springer Berlin Heidelberg, 1989.