Coalgebra course, lecture 8: Language equality with bisimulations up to congruence

Jurriaan Rot

October 17, 2016

1 Bisimulations and coinduction

Throughout this note we assume a fixed alphabet A. The set of words is denoted by A^*, the empty word by $\langle\rangle$ and the concatenation of words w and v by wv. We let $2 = \{0, 1\}$ be the set of Boolean values, where $0 \leq 1$. A language is a set of words, which we represent as a function from A^* to 2; the set of languages is denoted by 2^{A^*}. We abuse notation and use 0 and 1 to denote the empty language and the language containing only the empty word respectively. Further we let any alphabet letter $a \in A$ denote the language that contains only the letter itself.

A *(deterministic) automaton (DA)* over A is a triple (S, ϵ, δ) where S is a set of states, $\epsilon \colon S \to 2$ is an output function, and $\delta \colon S \to S^A$ is a transition function. A state $x \in S$ is *final* or *accepting* if $\epsilon(x) = 1$. We do not require S to be finite and fix no initial states.

Definition 1.1. Let (S, ϵ, δ) be a deterministic automaton. A relation $R \subseteq S \times S$ is a *bisimulation* if for any $(x, y) \in R$:

1. $\epsilon(x) = \epsilon(y)$, and
2. for all $a \in A \colon (\delta(x)(a), \delta(y)(a)) \in R$.

The largest bisimulation is denoted by $\sim$ and called *bisimilarity*; if $x \sim y$ then we say x is *bisimilar* to y.

We will instantiate the notion of bisimulation to a special deterministic automaton, whose state space is given by the set of languages 2^{A^*}; we call this the *automaton of languages* (over A). The output of a language L is simply $L(\langle\rangle)$, that is, a language is an accepting state precisely if it contains the empty word. We write $L\!\downarrow$ if this is the case. For any $a \in A$, the a-transition from a language L is given by *language derivative* L_a, defined as follows for all $w \in A^*$:

$$L_a(w) = L(aw)\,.$$

Spelling out Definition 1.1, a relation R on languages is a bisimulation on the automaton of languages if for any $(L, K) \in R$:

$$L\!\downarrow \text{ iff } K\!\downarrow, \text{ and for all } a \in A \colon (L_a, K_a) \in R\,.$$

Bisimilarity of languages is a characterization of language equality. This is called the *coinductive proof principle*, or simply coinduction.

Theorem 1.2 (Coinduction)**.** *For any two languages L, K:*

$$L \sim K \text{ iff } L = K\,.$$

Proof. For the implication from right to left, one shows that the diagonal relation $\{(L, L) \mid L \in 2^{A^*}\}$ is a bisimulation. For the converse, one can prove that for any languages L, K and any word w: if $L \sim K$ then $L(w) = K(w)$, by (structural) induction on w. □

The above coinduction principle is a concrete proof method: to show that two languages L, K are equal, it suffices to construct a bisimulation containing (L, K).

1.1 Regular operations

Consider the regular operations of union $L + K$, concatenation $L \cdot K$ (often written as LK) and Kleene star L^*. These are defined, for all w, as usual: $(L+K)(w) = L(w) \vee K(w)$, $(L \cdot K)(w) = 1$ iff there are v, u such that $w = vu$ and $L(v) = 1 = K(u)$, and $L^* = \sum_{i \geq 0} L^i$, where $L^0 = 1$ and $L^{i+1} = L \cdot L^i$. In order to prove equivalence of expressions involving the above operations, we may use bisimulations, but this requires a characterization of the output (acceptance of the empty word) and the derivatives of operations in terms of their arguments. Such a characterization was given for regular expressions by Brzozowski.

Lemma 1.3. *For any two languages L, K and for any $a, b \in A$:*

$$\begin{array}{ll}
\neg(0\!\downarrow) & 0_a = 0 \\
1\!\downarrow & 1_a = 0 \\
\neg(b\!\downarrow) & b_a = \begin{cases} 1 & \text{if } b = a \\ 0 & \text{otherwise} \end{cases} \\
(L + K)\!\downarrow \text{ iff } L\!\downarrow \vee K\!\downarrow & (L + K)_a = L_a + K_a \\
(L \cdot K)\!\downarrow \text{ iff } L\!\downarrow \wedge K\!\downarrow & (L \cdot K)_a = \begin{cases} L_a \cdot K + K_a & \text{if } L\!\downarrow \\ L_a \cdot K & \text{otherwise} \end{cases} \\
L^*\!\downarrow & (L^*)_a = L_a \cdot L^*
\end{array}$$

Example 1.4. We prove that $L \cdot 0 = L$ for all $L \in 2^{A^*}$. Consider the relation

$$R = \{(L \cdot 0, 0) \mid L \in 2^{A^*}\}$$

To show that R is a bisimulation, we first consider the output: we have $\neg(L \cdot 0\!\downarrow)$ and $\neg(0\!\downarrow)$, using Lemma 1.3. For any $a \in A$, if $L(\varepsilon) = 0$, we have

$$(L \cdot 0)_a = L_a \cdot 0 \; R \; 0 = 0_a$$

and if $\varepsilon \in L$, then

$$(L \cdot 0)_a = L_a \cdot 0 + 0_a = L_a \cdot 0 + 0 = L_a \cdot 0 \; R \; 0 = 0_a$$

We have shown that R is a bisimulation, so that $L \cdot 0 = L$ for every language $L \in 2^{A^*}$.

2 Bisimulation up-to for regular operations

In this section, we introduce an enhancement of the bisimulation proof method for equality of languages. We first illustrate the need for such an enhancement with a few examples. Consider the property $LL^* + 1 = L^*$. In order to prove this identity coinductively, we may try to show that

$$R = \{(LL^* + 1, L^*) \mid L \in 2^{A^*}\}$$

is a bisimulation. Using Lemma 1.3, it is easy to see that $(LL^* + 1)\downarrow$ iff $L^*\downarrow$ for any language L. Further, for any $a \in A$:

$$(LL^* + 1)_a = L_a L^* = (L^*)_a$$

where the leftmost and rightmost equality are follow from several applications of Lemma 1.3 and some standard identities between the regular languages (exercise!). Now we have shown that the derivatives are *equal*; this does not show that R is a bisimulation, since for that, the derivatives need to be *related by* R. The solution, however, is straightforward. If we augment the relation R as follows:

$$R' = R \cup \{(L, L) \mid L \in 2^{A^*}\}$$

then the derivatives of $LL^* + 1$ and L^* are related by R'; moreover, the diagonal is easily seen to satisfy the properties of a bisimulation as well. This solves the problem, but it is arguably somewhat inconvenient that additional work is required to deal with derivatives that are already equal.

As another motivating example, we consider the relation

$$R = \{(L^*L + 1, L^*) \mid L \in 2^{A^*}\}.$$

The derivatives are (using Lemma 1.3):

$$(L^*L + 1)_a = L_a L^* L + L_a + 0 = L_a(L^*L + 1) \qquad \text{and} \qquad (L^*)_a = L_a L^*$$

Clearly $L_a L^*$ can be obtained from $L_a(L^*L + 1)$ by replacing $L^*L + 1$ by L^*, and indeed the latter two languages are related by R. However, since these derivatives are not related *directly* by R, this argument does not show R to be a bisimulation. Extending R to an actual bisimulation is possible but requires a bit of work that one would rather skip.

To deal with examples such as the above in a more natural and easy way, we introduce the notion of *bisimulation up to congruence*. This requires the definition of congruence closure.

Definition 2.1. For a relation $R \subseteq 2^{A^*} \times 2^{A^*}$, define the *congruence closure* of R (with respect to $+$, $\cdot$ and *) as the least relation $\equiv$ satisfying the following rules:

$$\frac{L\,R\,K}{L \equiv K} \qquad \frac{}{L \equiv L} \qquad \frac{L \equiv K}{K \equiv L} \qquad \frac{L \equiv K \qquad K \equiv M}{L \equiv M}$$

$$\frac{L_1 \equiv K_1 \qquad L_2 \equiv K_2}{L_1 + L_2 \equiv K_1 + K_2} \qquad \frac{L_1 \equiv K_1 \qquad L_2 \equiv K_2}{L_1 \cdot L_2 \equiv K_1 \cdot K_2} \qquad \frac{L \equiv K}{L^* \equiv K^*}$$

In the sequel, we denote the congruence closure of a given relation R on languages by $\equiv_R$, or $\equiv$ if R is clear from the context.

The first rule ensures that $R \subseteq \equiv_R$. The three rules on the right in the first row turn $\equiv_R$ into an equivalence relation. The three rules on the second row ensure that $\equiv_R$ is closed under the operations under consideration, which in particular means that $\equiv_R$ relates languages obtained by (syntactic) substitution of languages related by R. For example, if $(L^*L + 1, L^*) \in R$, then we can derive from the above rules that $L_a(L^*L + 1) \equiv_R L_aL^*$.

Definition 2.2. A relation $R \subseteq 2^{A^*} \times 2^{A^*}$ is a *bisimulation up to congruence*, or simply a *bisimulation up-to*, if for any pair $(L, K) \in R$:

1. $L\!\downarrow$ iff $K\!\downarrow$, and
2. for all $a \in A$: $L_a \equiv_R K_a$.

In a bisimulation up to congruence, the derivatives can be related by the congruence $\equiv_R$ rather than the relation R itself, and therefore, bisimulations up-to may be easier to construct than bisimulations. Indeed, to prove that R is a bisimulation up-to, the derivatives can be related by familiar equational reasoning.

A bisimulation up-to is, in general, not a bisimulation. However, as we show below, it *represents* one, in the following sense: if R is a bisimulation up-to, then $\equiv_R$ is a bisimulation.

Theorem 2.3. *If R is a bisimulation up to congruence then for any $(L, K) \in R$, we have $L = K$.*

Proof. Let $\equiv$ be the congruence closure of R. We show that any pair (L, K) in $\equiv$ satisfies the properties

1. $L\!\downarrow$ iff $K\!\downarrow$ and
2. for any $a \in A$: $L_a \equiv K_a$

of a bisimulation, by rule induction on $(L, K) \in \equiv$. This amounts to showing that $\equiv$ is closed under the inference rules of Definition 2.2. For the base cases:

1. for the pairs contained in R, the conditions are satisfied by the assumption that R is a bisimulation up-to;
2. the case $L \equiv L$ is trivial.

Now assume languages L, K, M, N such that $L \equiv K$, $M \equiv N$, $L\!\downarrow$ iff $K\!\downarrow$, $M\!\downarrow$ iff $N\!\downarrow$ and for all $a \in A$: $L_a \equiv K_a$ and $M_a \equiv N_a$. We need to prove that $(L+M, K+N), (LM, KN), (L^*, K^*)$, (K, L) and (L, N) (if $K = M$) again satisfy the properties of a bisimulation, i.e., $(L + M)\!\downarrow$ iff $(K + N)\!\downarrow$ and for all $a \in A$: $(L + M)_a \equiv (K + N)_a$, and similarly for the other operations. We treat the case of union: $(L + M)\!\downarrow$ iff $L\!\downarrow \vee M\!\downarrow$ iff $K\!\downarrow \vee N\!\downarrow$ iff $(K + N)\!\downarrow$; moreover by assumption and closure of $\equiv$ under $+$ we have $L_a + M_a \equiv K_a + N_a$, and so $(L + M)_a = L_a + M_a \equiv K_a + N_a = (K + N)_a$.

Concatenation and Kleene star are treated in a similar manner, and symmetry and transitivity are not difficult either. Thus, by induction, $\equiv$ is a bisimulation, so by Theorem 1.2 we have $L = K$ for any $L \equiv K$ and for any $(L, K) \in R$, in particular. □

Any bisimulation is also a bisimulation up-to, so Theorem 2.3 is a generalization of Theorem 1.2 for the case of languages. Consequently, its converse holds as well.

We proceed with a number of proofs based on bisimulation up-to.

Example 2.4. Recall the relation $R = \{(L^*L + 1, L^*) \mid L \in 2^{A^*}\}$ from the beginning of this section. As we have seen, the a-derivatives are $L_a(L^*L + 1)$ and L_aL^*, which are not related by R; however they *are* related by $\equiv_R$. So R is a bisimulation up-to, and consequently $L^*L + 1 = L^*$, by Theorem 2.3. Moreover, the relation $\{(LL^* + 1, L^*) \mid L \in 2^{A^*}\}$ from the beginning of this section is a bisimulation up-to as well; there, the derivatives are equal and thus related by $\equiv_R$.

Example 2.5. Arden's rule states that if $L = KL + M$ for some languages L, K and M, and K does not contain the empty word, then $L = K^*M$. In order to prove its validity coinductively, let L, K, M be languages such that $\neg(K\!\downarrow)$ and $L = KL + M$, and let $R = \{(L, K^*M)\}$. Using that $\neg(K\!\downarrow)$, we have $L\!\downarrow$ iff $(KL + M)\!\downarrow$ iff $M\!\downarrow$ iff $K^*M\!\downarrow$. Further,

$$L_a = (KL + M)_a = K_aL + M_a \equiv_R K_aK^*M + M_a = (K^*M)_a$$

for any $a \in A$. So R is a bisimulation up-to, proving Arden's rule.