Bisimulation equivalence of first-order grammars is Ackermann-hard

Petr Jančar

Dept Comp. Sci., FEI, Techn. Univ. Ostrava, Czech Republic,
http://www.cs.vsb.cz/jancar/,
email: petr.jancar@vsb.cz

Abstract

Bisimulation equivalence (or bisimilarity) of first-order grammars is decidable, as follows from the decidability result by Sénizergues (1998, 2005) that has been given in an equivalent framework of equational graphs with finite out-degree, or of pushdown automata (PDA) with only deterministic and popping ε-transitions. Benedikt, Göller, Kiefer, and Murawski (2013) have shown that the bisimilarity problem for PDA (even) without ε-transitions is nonelementary. Here we show Ackermann-hardness for bisimilarity of first-order grammars. The grammars do not use explicit ε-transitions, but they correspond to the above mentioned PDA with (deterministic and popping) ε-transitions, and this feature is substantial in the presented lower-bound proof. The proof is based on a (polynomial) reduction from the reachability problem of reset (or lossy) counter machines, for which the Ackermann-hardness has been shown by Schnoebelen (2010); in fact, this reachability problem is known to be Ackermann-complete, i.e., $\mathbf{F}_\omega$-complete in the hierarchy of fast-growing complexity classes defined by Schmitz (2013).

Keywords:
first-order grammar, term rewriting, pushdown automaton, bisimulation, complexity

Introduction

Bisimulation equivalence, also called bisimilarity, has been recognized as a fundamental behavioural equivalence of systems. It is thus natural to explore the related decidability and complexity questions for various computational models.

The involved result by Sénizergues [7], showing the decidability of bisimilarity for equational graphs of finite out-degree, or equivalently for pushdown automata (PDA) with only deterministic and popping ε-transitions, surely belongs to the most fundamental results in this area. (This result generalized Sénizergues's solving of the famous DPDA equivalence problem.) The complexity of the bisimilarity problem has been recently shown to be nonelementary, by Benedikt, Göller, Kiefer, and Murawski [1], even for PDA with no ε-transitions.

Here we look at the bisimilarity problem in the framework of first-order grammars (i.e., finite sets of term root-rewriting rules). This framework is long known as equivalent to the PDA framework. (We can refer, e.g., to [3] for a recent use of a concrete respective transformation.) Hence Sénizergues's decidability proof applies to the grammars as well. We show that the problem for grammars is "Ackermann-hard", and thus not primitive recursive. Though

the grammars do not use explicit ε-transitions, they correspond to the above mentioned PDA with deterministic and popping ε-transitions, and this feature is substantial in the presented lower-bound proof. Hence the nonelementary bound of [1] remains the best known lower bound for bisimilarity of pushdown systems without ε-transitions.

The presented proof is based on a (polynomial) reduction from the reachability problem of reset (or lossy) counter machines, for which the Ackermann-hardness has been shown by Schnoebelen (see [5] and the reference therein). In fact, we also know an "Ackermannian" upper bound for this reachability problem [2], and the problem is thus $\mathbf{F}_\omega$-complete (or Ack-complete) in the hierarchy of fast-growing complexity classes defined by Schmitz [6]. The question of a similar upper bound for the bisimilarity problem is not addressed in this paper.

Definitions, and the result

We briefly recall the notions that are needed for stating the result. We use the forms that are convenient here; e.g., we (harmlessly) define bisimulations as symmetric, though this is usually not required.

First-order terms. We assume a fixed set of *variables* $\textsc{Var} = \{x_1, x_2, \dots\}$. Given a set $\mathcal{N}$ of function symbols with arities, by $\textsc{Terms}_{\mathcal{N}}$ we denote the set of *terms over* $\mathcal{N}$. A *term* $T \in \textsc{Terms}_{\mathcal{N}}$ is either a variable x_i or $A(U_1, \dots, U_m)$ where $A \in \mathcal{N}$, $\mathit{arity}(A) = m$, and $U_1, \dots, U_m$ are terms.

First-order grammars. A (first-order) *grammar* is a tuple $\mathcal{G} = (\mathcal{N}, \Sigma, \mathcal{R})$ where $\mathcal{N}$ is a finite set of ranked *nonterminals* (or function symbols with arities), Σ is a finite set of *actions* (or terminals), and $\mathcal{R}$ is a finite set of (root-rewriting) *rules* of the form $A(x_1, \dots, x_m) \xrightarrow{a} V$ where $A \in \mathcal{N}$, $\mathit{arity}(A) = m$, $a \in \Sigma$, and $V \in \textsc{Terms}_{\mathcal{N}}$ is a term in which all occurring variables are from the set $\{x_1, \dots, x_m\}$. (Some example rules are $A(x_1, x_2, x_3) \xrightarrow{b} C(D(x_3, B), x_2)$, $A(x_1, x_2, x_3) \xrightarrow{b} x_2$, $D(x_1, x_2) \xrightarrow{a} A(D(x_2, x_2), x_1, B)$; here the arities of A, B, C, D are $3, 0, 2, 2$, respectively.)

LTSs associated with grammars. A grammar $\mathcal{G} = (\mathcal{N}, \Sigma, \mathcal{R})$ defines the *labelled transition system* $\mathcal{L}_{\mathcal{G}} = (\textsc{Terms}_{\mathcal{N}}, \Sigma, (\xrightarrow{a})_{a \in \Sigma})$ in which each rule $A(x_1, \dots, x_m) \xrightarrow{a} V$ from $\mathcal{R}$ induces transitions $(A(x_1, \dots, x_m))\sigma \xrightarrow{a} V\sigma$ for all substitutions $\sigma : \textsc{Var} \to \textsc{Terms}_{\mathcal{N}}$. (The above example rules thus induce, e.g., $A(x_1, x_2, x_3) \xrightarrow{b} C(D(x_3, B), x_2)$ (here $\sigma(x_i) = x_i$), $A(V, x_5, U) \xrightarrow{b} C(D(U, B), x_5)$ (here $\sigma(x_1) = V$, $\sigma(x_2) = x_5$, $\sigma(x_3) = U$), $A(U_1, U_2, U_3) \xrightarrow{b} C(D(U_3, B), U_2)$, $A(U_1, U_2, U_3) \xrightarrow{b} U_2$, etc.)

Bisimulation equivalence. Given $\mathcal{G} = (\mathcal{N}, \Sigma, \mathcal{R})$, a set (or a relation) $\mathcal{B} \subseteq \textsc{Terms}_{\mathcal{N}} \times \textsc{Terms}_{\mathcal{N}}$ is a *bisimulation* if it is *symmetric* ($(T, U) \in \mathcal{B}$ implies $(U, T) \in \mathcal{B}$), and for any $(T, U) \in \mathcal{B}$ and $T \xrightarrow{a} T'$ there is U' such that $U \xrightarrow{a} U'$ and $(T', U') \in \mathcal{B}$. Two *terms* T, U are *bisimilar*, written $T \sim U$, if there is a bisimulation containing (T, U).
The *bisimilarity problem for first-order grammars* asks, given a grammar $\mathcal{G}$ and terms T, U, whether $T \sim U$.

Ackermann-hardness. We refer to [6] for detailed definitions of the class $\mathbf{F}_\omega$ (or Ack) of decision problems, and of the problems that are $\mathbf{F}_\omega$-complete or $\mathbf{F}_\omega$-hard. Here we just recall the "Ackermannian" function $f_A : \mathbb{N} \to \mathbb{N}$ (where $\mathbb{N} = \{0, 1, 2, \dots\}$) defined as follows: we first define the family $f_0, f_1, f_2, \dots$ by putting $f_0(n) = n{+}1$ and $f_{k+1}(n) = f_k(f_k(\dots f_k(n) \dots))$ where f_k is applied $n{+}1$ times; then we put $f_A(n) = f_n(n)$. The problem $\mathrm{HP}_{\mathrm{Ack}}$ that asks, given a Turing machine M, an input w, and some $n \in \mathbb{N}$, whether M halts on w within $f_A(n)$ steps, is an example of an *Ackermann-complete problem*. We say (in this paper) that a

problem $\mathcal{P}$ is *Ackermann-hard* if $\mathrm{HP}_{\mathrm{ACK}}$ is reducible to $\mathcal{P}$, or to the complementary problem co-$\mathcal{P}$, by a standard polynomial many-one reduction. (The notion is more general in [6], and it also includes primitive-recursive reductions.)

Theorem 1. *The bisimilarity problem for first-order grammars is Ackermann-hard.*

Proof of the theorem

A direct reduction from $\mathrm{HP}_{\mathrm{ACK}}$ would require many technicalities. Fortunately, these have been already handled in deriving Ackermann-hardness (or Ackermann-completeness) of other problems. For our reduction we thus choose a more convenient Ackermann-hard problem (which is Ackermann-complete, in fact), namely the reachability problem for reset counter machines.

Reset Counter Machines (RCMs). An *RCM* is a tuple $\mathcal{M} = (d, Q, \delta)$ where d is the *dimension*, yielding d nonnegative *counters* $c_1, c_2, \ldots, c_d$, Q is a finite set of *(control) states*, and $\delta \subseteq Q \times \textsc{Op} \times Q$ is a finite set of *instructions*, where the set $\textsc{Op}$ of *operations* contains $\textsc{inc}(c_i)$ (increment c_i), $\textsc{dec}(c_i)$ (decrement c_i), and $\textsc{reset}(c_i)$ (set c_i to 0), for $i = 1, 2, \ldots, d$. We view $Q \times \mathbb{N}^d$ as the set $\textsc{Conf}$ of *configurations* of $\mathcal{M}$. The *transition relation* $\longrightarrow \subseteq \textsc{Conf} \times \textsc{Conf}$ is induced by δ in the obvious way: If $(p, op, q) \in \delta$ then we have $(p, (n_1, \ldots, n_d)) \longrightarrow (q, (n'_1, \ldots, n'_d))$ in the following cases:

- $op = \textsc{inc}(c_i)$, $n'_i = n_i{+}1$, and $n'_j = n_j$ for all $j \neq i$; or
- $op = \textsc{dec}(c_i)$, $n_i > 0$, $n'_i = n_i{-}1$, and $n'_j = n_j$ for all $j \neq i$; or
- $op = \textsc{reset}(c_i)$, $n'_i = 0$, and $n'_j = n_j$ for all $j \neq i$.

By $\longrightarrow^*$ we denote the reflexive and transitive closure of $\longrightarrow$.

Complexity of the reachability problem for RCMs. We define the *RCM-reachability problem* in the following convenient form: given an RCM $\mathcal{M} = (d, Q, \delta)$ and (control) states p_{IN}, p_{F}, we ask if p_{F} is reachable from $(p_{\mathrm{IN}}, (0, 0, \ldots, 0))$, i.e., if there are $m_1, m_2, \ldots, m_d \in \mathbb{N}$ such that $(p_{\mathrm{IN}}, (0, 0, \ldots, 0)) \longrightarrow^* (p_{\mathrm{F}}, (m_1, m_2, \ldots, m_d))$. The known results yield:

Theorem 2. *RCM-reachability problem is Ackermann-complete.*

It is the lower bound, the Ackermann-hardness, which is important for us here; we refer to [5] for a proof and further references. (See also [8] for an independent proof related to relevance logic.) The upper bound follows from [2]. Hence the problem is $\mathbf{F}_\omega$-complete in the sense of [6]. We note that the same result holds for *lossy counter machines*; they have the zero-test instead of the reset, and any counter can spontaneously decrease at any time. (We prefer RCMs since they are a slightly simpler model.)

RCM-reachability reduces to first-order bisimilarity. We finish by proving the next lemma; this establishes Theorem 1, by using the hardness part of Theorem 2. The reduction in the proof of the lemma is obviously polynomial; in fact, it can be checked to be a logspace reduction, but this is a minor point in the view of the fact that even a primitive-recursive reduction would suffice here.

Lemma 3. *The RCM-reachability problem is polynomially reducible to the complement of the bisimilarity problem for first-order grammars.*

Proof. Let us consider an instance $\mathcal{M} = (d, Q, \delta)$, p_{IN}, p_{F} of the RCM-reachability problem, and imagine the following game between Attacker (he) and Defender (she). This is the first version of a game that will be afterwards implemented as a standard bisimulation game. Attacker aims to show that p_{F} is reachable from $(p_{\text{IN}}, (0, 0, \ldots, 0))$, while Defender opposes this.

The game uses $2d$ *game-counters*, which are never decremented; each counter c_i of $\mathcal{M}$ yields two game-counters, namely c_i^I and c_i^D, for counting the numbers of Increments and Decrements of c_i, respectively, since the last reset or since the beginning if there has been no reset of c_i so far. The *initial position* is $(p_{\text{IN}}, ((0, 0), \ldots, (0, 0)))$, with all $2d$ game-counters (organized in pairs) having the value 0.

A game *round* from position $(p, ((n_1, n_1'), \ldots, (n_d, n_d')))$ proceeds as described below. It will become clear that it suffices to consider only the cases $n_i \geq n_i'$; the position then corresponds to the $\mathcal{M}$'s configuration $(p, (n_1 - n_1', \ldots, n_d - n_d'))$.

If $p = p_{\text{F}}$, then Attacker wins; if $p \neq p_{\text{F}}$ and there is no instruction $(p, op, q) \in \delta$, then Defender wins. Otherwise Attacker chooses $(p, op, q) \in \delta$, and the continuation depends on op:

1. If $op = \text{INC}(c_i)$, then the next-round position arises (from the previous one) by replacing p with q and by performing $c_i^I := c_i^I + 1$ (the counter of increments of c_i is incremented, i.e., n_i is replaced with $n_i + 1$).

2. If $op = \text{RESET}(c_i)$, then the next-round position arises by replacing p with q and by performing $c_i^I := 0$ and $c_i^D := 0$ (hence both n_i and n_i' are replaced with 0).

3. If $op = \text{DEC}(c_i)$, then *Defender chooses* one of the following options:

   (a) the next-round position arises by replacing p with q and by performing $c_i^D := c_i^D + 1$ (the counter of decrements of c_i is incremented, i.e., n_i' is replaced with $n_i' + 1$), or

   (b) (Defender claims that this decrement is *illegal* since $n_i = n_i'$ and) the next position becomes just (n_i, n_i'). In this case a (deterministic) check if $n_i = n_i'$ is performed, by successive synchronized decrements at both sides. If indeed $n_i = n_i'$ (the counter-bottoms are reached at the same moment), then Defender wins; otherwise (when $n_i \neq n_i'$) Attacker wins.

If $(p_{\text{IN}}, (0, 0, \ldots, 0)) \longrightarrow^* (p_{\text{F}}, (m_1, m_2, \ldots, m_d))$ for some $m_1, m_2, \ldots, m_d$, i.e., if the answer to RCM-reachability is YES, then Attacker has a winning strategy: he just follows the corresponding sequence of instructions. He thus also always chooses $\text{DEC}(c_i)$ *legally*, i.e. only in the cases where $n_i > n_i'$, and Defender loses if she ever chooses 3(b). If the answer is NO (p_{F} is not reachable), and Attacker follows a legal sequence of instructions, then he either loses in a "dead" state or the play is infinite; if Attacker chooses an illegal decrement, then in the first such situation we obviously have $n_i = n_i'$ for the respective counter c_i, and Defender can force her win via 3(b).

Since the game-counters can be only incremented or reset, it is a routine to implement the above game as a bisimulation game in the grammar-framework (using a standard method of "Defender's forcing" for implementing the choice in 3). We now describe the corresponding grammar $\mathcal{G} = (\mathcal{N}, \Sigma, \mathcal{R})$.

The set $\mathcal{N}$ of nonterminals will include a unary nonterminal I, a nullary nonterminal $\perp$, and the nonterminals with arity $2d$ that are induced by control states of $\mathcal{M}$ as follows: each $p \in Q$ induces $A_p, A_{(p,i)}, B_p, B_{(p,i,1)}, B_{(p,i,2)}$, where $i = 1, 2, \ldots, d$.

We intend that a game-position $(p, ((n_1, n'_1), \ldots, (n_d, n'_d)))$ corresponds to the pair of terms

$$\Big(A_p(I^{n_1}\bot, I^{n'_1}\bot, \ldots, I^{n_d}\bot, I^{n'_d}\bot),\ B_p(I^{n_1}\bot, I^{n'_1}\bot, \ldots, I^{n_d}\bot, I^{n'_d}\bot)\Big) \tag{1}$$

where $I^k\bot$ is a shorthand for $I(I(\ldots I(\bot)\ldots))$ with I occurring k times; we put $I^0\bot = \bot$. The RCM-reachability instance $\mathcal{M}, p_{\text{IN}}, p_{\text{F}}$ will be reduced to the (non)bisimilarity-problem instance $\mathcal{G}$, $A_{p_{\text{IN}}}(\bot, \ldots, \bot)$, $B_{p_{\text{IN}}}(\bot, \ldots, \bot)$.

We put $\Sigma = \delta \uplus \{a, b\}$, i.e., the actions of $\mathcal{G}$ correspond to the instructions (or instruction names) of $\mathcal{M}$, and we also use auxiliary actions a, b.

The set of rules $\mathcal{R}$ contains a sole rule for I, namely $I(x_1) \xrightarrow{a} x_1$, and no rule for $\bot$; hence $I^n\bot \sim I^{n'}\bot$ iff $n = n'$. Each instruction $\textsc{Ins} = (p, op, q) \in \delta$ induces the rules in $\mathcal{R}$ as follows:

1. If $op = \textsc{inc}(c_i)$, then the induced rules are
   $A_p(x_1, \ldots, x_{2d}) \xrightarrow{\textsc{Ins}} A_q(x_1, \ldots, x_{2(i-1)}, I(x_{2i-1}), x_{2i}, \ldots, x_{2d})$, and
   $B_p(x_1, \ldots, x_{2d}) \xrightarrow{\textsc{Ins}} B_q(x_1, \ldots, x_{2(i-1)}, I(x_{2i-1}), x_{2i}, \ldots, x_{2d})$.

2. If $op = \textsc{reset}(c_i)$, then the induced rules are
   $A_p(x_1, \ldots, x_{2d}) \xrightarrow{\textsc{Ins}} A_q(x_1, \ldots, x_{2(i-1)}, \bot, \bot, x_{2i+1}, \ldots, x_{2d})$,
   $B_p(x_1, \ldots, x_{2d}) \xrightarrow{\textsc{Ins}} B_q(x_1, \ldots, x_{2(i-1)}, \bot, \bot, x_{2i+1}, \ldots, x_{2d})$.

3. If $op = \textsc{dec}(c_i)$, then the induced rules are below; here we use the shorthand $A \xrightarrow{a} B$ when meaning $A(x_1, \ldots, x_{2d}) \xrightarrow{a} B(x_1, \ldots, x_{2d})$:
   $A_p \xrightarrow{\textsc{Ins}} A_{(q,i)}$, $A_p \xrightarrow{\textsc{Ins}} B_{(q,i,1)}$, $A_p \xrightarrow{\textsc{Ins}} B_{(q,i,2)}$, $B_p \xrightarrow{\textsc{Ins}} B_{(q,i,1)}$, $B_p \xrightarrow{\textsc{Ins}} B_{(q,i,2)}$,
   $A_{(q,i)}(x_1, \ldots, x_{2d}) \xrightarrow{a} A_q(x_1, \ldots, x_{2i-1}, I(x_{2i}), x_{2i+1}, \ldots, x_{2d})$,
   $B_{(q,i,1)}(x_1, \ldots, x_{2d}) \xrightarrow{a} B_q(x_1, \ldots, x_{2i-1}, I(x_{2i}), x_{2i+1}, \ldots, x_{2d})$,
   $B_{(q,i,2)}(x_1, \ldots, x_{2d}) \xrightarrow{a} A_q(x_1, \ldots, x_{2i-1}, I(x_{2i}), x_{2i+1}, \ldots, x_{2d})$,
   $A_{(q,i)}(x_1, \ldots, x_{2d}) \xrightarrow{b} x_{2i-1}$, $B_{(q,i,1)}(x_1, \ldots, x_{2d}) \xrightarrow{b} x_{2i-1}$, $B_{(q,i,2)}(x_1, \ldots, x_{2d}) \xrightarrow{b} x_{2i}$.

Moreover, $\mathcal{R}$ will also contain $A_{p_{\text{F}}}(x_1, \ldots, x_{2d}) \xrightarrow{a} \bot$ (but *not* $B_{p_{\text{F}}}(x_1, \ldots, x_{2d}) \xrightarrow{a} \bot$).

Now we recall the standard (turn-based) *bisimulation game*, starting with the pair $(A_{p_{\text{IN}}}(\bot, \ldots, \bot), B_{p_{\text{IN}}}(\bot, \ldots, \bot))$. In the round starting with (T_1, T_2), Attacker chooses a transition $T_j \xrightarrow{a} T'_j$ and then Defender chooses $T_{3-j} \xrightarrow{a} T'_{3-j}$ (for the same $a \in \Sigma$); the next round starts with the pair (T'_1, T'_2). If a player gets stuck, then (s)he loses; an infinite play is a win of Defender. It is obvious that Defender has a winning strategy in this game iff $A_{p_{\text{IN}}}(\bot, \ldots, \bot) \sim B_{p_{\text{IN}}}(\bot, \ldots, \bot)$.

We now easily check that this bisimulation game indeed implements the above described game; a game-position $(p, ((n_1, n'_1), \ldots, (n_d, n'_d)))$ is implemented as the pair (1). The points 1 and 2 directly correspond to the previous points 1 and 2. If Attacker chooses an instruction $\textsc{Ins} = (p, \textsc{dec}(c_i), q)$, then he must use the respective rule $A_p \xrightarrow{\textsc{Ins}} A_{(q,i)}$ in 3, since otherwise Defender installs syntactic equality, i.e. a pair (T, T). It is now Defender who chooses $B_p \xrightarrow{\textsc{Ins}} B_{(q,i,1)}$ (corresponding to the previous 3(a)) or $B_p \xrightarrow{\textsc{Ins}} B_{(q,i,2)}$ (corresponding to 3(b)). Attacker then must choose action a in the first case, and action b in the second case; otherwise we get syntactic equality. The first case thus results in the pair $(A_q(\ldots), B_q(\ldots))$

corresponding to the next game-position (where c_i^D has been incremented), and the second case results in the pair $(I^{n_i}\bot, I^{n'_i}\bot)$; we have already observed that $I^{n_i}\bot \sim I^{n'_i}\bot$ iff $n_i = n'_i$.

Finally we observe that in any pair $(A_{p_{\rm F}}(\dots), B_{p_{\rm F}}(\dots))$ Attacker wins immediately (since the transition $A_{p_{\rm F}}(\dots) \stackrel{a}{\longrightarrow} \bot$ can not be matched).

We have thus established that $p_{\rm F}$ is reachable from $(p_{\rm IN}, (0, \dots, 0))$ if, and only if, $A_{p_{\rm IN}}(\bot, \dots, \bot) \not\sim B_{p_{\rm IN}}(\bot, \dots, \bot)$. □

Additional remarks

In the above bisimulation game we obviously encounter "unbalanced" terms; the syntactic tree of an unbalanced term has branches of different lengths. This feature is related to deterministic popping ε-transitions in the pushdown automata corresponding to our grammars. Hence the nonelementary bound shown in [1] remains the best known lower bound for bisimilarity of pushdown systems without ε-transitions.

We can add that introducing (popping) ε-rules $A(x_1, \dots, x_m) \stackrel{\varepsilon}{\longrightarrow} x_i$ in our grammars (where such a rule might be not the only one with $A(x_1, \dots, x_m)$ at the left-hand side) already yields undecidability of bisimilarity [4].

Author's acknowledgement

The reported result has been achieved during my visit at LSV ENS Cachan, and I am grateful to Sylvain Schmitz and Philippe Schnoebelen for fruitful discussions.

References

[1] M. Benedikt, S. Göller, S. Kiefer, and A. S. Murawski. Bisimilarity of pushdown automata is nonelementary. In *Proc. LICS 2013*, pages 488–498. IEEE Computer Society, 2013.

[2] D. Figueira, S. Figueira, S. Schmitz, and Ph. Schnoebelen. Ackermannian and primitive-recursive bounds with Dickson's lemma. In *Proc. LICS 2011*, pages 269–278. IEEE Computer Society, 2011.

[3] P. Jančar. Decidability of DPDA language equivalence via first-order grammars. In *Proc. LICS 2012*, pages 415–424. IEEE Computer Society, 2012.

[4] P. Jančar and J. Srba. Undecidability of bisimilarity by Defender's forcing. *J. ACM*, 55(1), 2008.

[5] Ph. Schnoebelen. Revisiting Ackermann-hardness for lossy counter machines and reset Petri nets. In *Proc. MFCS 2010*, volume 6281 of *Lecture Notes in Computer Science*, pages 616–628. Springer, 2010.

[6] S. Schmitz. Complexity hierarchies beyond elementary. Manuscript, 2013.

[7] G. Sénizergues. The bisimulation problem for equational graphs of finite out-degree. *SIAM J.Comput.*, 34(5):1025–1106, 2005. (A preliminary version appeared at FOCS'98.).

[8] A. Urquhart. The complexity of decision procedures in relevance logic II. *J. Symb. Log.*, 64(4):1774–1802, 1999.