

# Deciding $\mathcal{H}_1$ by Resolution <sup>★</sup>

Jean Goubault-Larrecq <sup>a,\*</sup>

<sup>a</sup>*LSV/CNRS UMR 8643 & INRIA Futurs projet SECSI & ENS Cachan  
61, av. du président-Wilson, 94235 Cachan Cedex, France*

---

## Abstract

Nielson, Nielson and Seidl's class  $\mathcal{H}_1$  is a decidable class of first-order Horn clause sets, describing strongly regular relations. We give another proof of decidability, and of the regularity of the defined languages, based on fairly standard automated deduction techniques.

*Key words:* Automatic theorem proving, formal languages.

---

Nielson et al. [6] introduced the class  $\mathcal{H}_1$  of first-order Horn clause sets to model reachability in the spi-calculus. They showed that  $\mathcal{H}_1$  satisfiability is decidable and DEXPTIME-complete, and that  $\mathcal{H}_1$  clause sets can be converted to equivalent tree automata in exponential time—so  $\mathcal{H}_1$  defines regular tree languages. Further subclasses of  $\mathcal{H}_1$ , namely  $\mathcal{H}_2$  and  $\mathcal{H}_3$ , have polynomial time complexity.

Our objective is to make it clear that fairly standard automated deduction techniques yield streamlined proofs of these facts, and in passing to introduce a slightly simpler definition of  $\mathcal{H}_1$ . We reprove most results of Nielson et al. [6] this way, sometimes correcting small inaccuracies.

**$\mathcal{H}_1$  and  $\mathcal{H}_1$ .** As usual, we fix a first-order signature, which we shall leave implicit. Terms are denoted  $s, t, u, v, \dots$ , predicate symbols  $P, Q, \dots$ , variables  $X, Y, Z, \dots$ . We assume there are finitely many predicate symbols. Horn clauses  $C$  are of the form  $H \Leftarrow \mathcal{B}$  where the *head*  $H$  is either an atom or  $\perp$ , and the *body*  $\mathcal{B}$  is a finite set  $A_1, \dots, A_n$  of atoms. If  $\mathcal{B}$  is empty ( $n = 0$ ), then  $C = H$  is a *fact*.

**Definition 1** *An  $\mathcal{H}_1$  clause is any Horn clause  $H \Leftarrow P_1(t_1), \dots, P_n(t_n)$ , where all predicate symbols are unary,  $t_1, \dots, t_n$  are arbitrary, and the head  $H$  is either  $\perp$ , of the form  $P(X)$  or of the form  $P(f(X_1, \dots, X_k))$  where  $X_1, \dots, X_k$  are distinct.*

---

<sup>★</sup> Partially supported by the RNTL projects EVA and Prouvé, the ACI SI Rossignol, and the ACI jeunes chercheurs “Sécurité informatique, proto. crypto. et détection d'intrusions”.

\* Fax: +33-1 47 40 75 21.

*Email address:* goubault@lsv.ens-cachan.fr (Jean Goubault-Larrecq).

*URL:* www.lsv.ens-cachan.fr/~goubault/ (Jean Goubault-Larrecq).

This is fairly unrestricted: apart from the use of unary predicates, which is innocuous—encode  $k$ -ary relations  $P(t_1, \dots, t_k)$  as  $P(c(t_1, \dots, t_k))$ —, the only restrictions on  ${}_b\mathcal{H}_1$  clauses are on heads. Let us restate Nielson et al. [6]’s definition. We depart from op.cit. only in that we only use unary predicates, to ease comparison. It should be clear that we do not lose any generality this way.

**Definition 2** An  $\mathcal{H}_1$  clause is any Horn clause  $H \Leftarrow P_1(t_1), \dots, P_n(t_n)$  where all predicate symbols are unary,  $t_1, \dots, t_n$  are arbitrary, and: (a) Its head  $H$  is linear, i.e., no variable occurs twice in  $H$ ; and (b) any two variables  $X, Y$  that are connected in the body and occur in  $H$  must be siblings in  $H$ .

This requires auxiliary definitions: given a Horn clause as above, let  $\cong$  (“is connected to”) be the smallest equivalence relation such that  $X \cong Y$  if both  $X$  and  $Y$  occur in the same  $P_i(t_i)$  for some  $i$ ,  $1 \leq i \leq n$ ;  $X$  and  $Y$  are *siblings* in  $H$  iff there is a subterm  $f(t_1, \dots, t_n)$  of  $H$  such that  $X = t_i$  and  $Y = t_j$  for some  $i, j$ ,  $1 \leq i, j \leq n$ . Every  ${}_b\mathcal{H}_1$  clause is in  $\mathcal{H}_1$ ; Proposition 4 below is a form of converse.

The main value of  ${}_b\mathcal{H}_1$  to us is that it provides a straightforward descriptive typing discipline for Horn clause sets, à la Frühwirth et al. [3]. Define the rewrite relation  $\rightsquigarrow$  on clause sets by (we use the letter  $\mathcal{B}$  to denote finite sets of atoms, and  $\mathcal{B}\{X := t\}$  to denote  $\mathcal{B}$  with  $X$  substituted for  $t$ ):

$$P(\mathcal{C}[t]) \Leftarrow \mathcal{B} \rightsquigarrow \begin{cases} P(\mathcal{C}[Z]) \Leftarrow \mathcal{B}, Q(Z) & (Z \text{ fresh}) \\ Q(t) \Leftarrow \mathcal{B} \end{cases} \quad (1)$$

where  $t$  is not a variable,  $Q$  is a fresh predicate symbol, and  $\mathcal{C}[]$  is a non-trivial one-hole context

$$P(\mathcal{C}[X]) \Leftarrow \mathcal{B} \rightsquigarrow P(\mathcal{C}[Y]) \Leftarrow \mathcal{B}, \mathcal{B}\{X := Y\} \quad (2)$$

where  $X$  occurs at least twice in  $\mathcal{C}[X]$ ,  $Y$  is a fresh variable

A one-hole context  $\mathcal{C}[]$  is a term with a distinguished occurrence of the *hole*  $[]$ .  $\mathcal{C}[u]$  is  $\mathcal{C}[]$  with  $u$  in place of the hole.  $\mathcal{C}[]$  is *non-trivial* iff  $\mathcal{C}[] \neq []$ . In (2), we require  $X$  to occur at least twice, with one occurrence distinguished by the hole in  $\mathcal{C}[]$ .

**Proposition 3** Starting from a clause set  $S_0$ ,  $\rightsquigarrow$  terminates in polynomially many steps. Any  $\rightsquigarrow$ -normal form of  $S_0$  is an  ${}_b\mathcal{H}_1$  clause set  $S_*$  that logically implies  $S_0$ .

**PROOF.** Let  $|t|$  be defined by  $|X| = 0$ ,  $|f(t_1, \dots, t_n)| = |t_1| + \dots + |t_n| + 1$ . Define the *defect*  $\partial t$  of  $t$  as  $|t| - 1 = |t_1| + \dots + |t_n|$  if  $t = f(t_1, \dots, t_n)$ , 0 otherwise. The *defect* of an atom  $P(t)$  is  $\partial t$ , that of  $\perp$  is 0. The *defect*  $\partial C$  of a clause  $C$  is that of its head. The *defect*  $\partial S$  of a Horn clause set is  $\sum_{C \in S} \partial C$ . Clearly  $\partial \mathcal{C}[t] = \partial \mathcal{C}[Z] + |t|$  when  $\mathcal{C}[u]$  is a non-trivial one-hole context, so  $\partial \mathcal{C}[Z] + \partial t = \partial \mathcal{C}[Z] + |t| - 1 < \partial \mathcal{C}[t]$  when  $t$  is not a variable; it follows that (1) makes  $\partial S$  decrease strictly, while (2) leaves it invariant. So in any rewrite sequence from  $S_0$ , (1) is applied at most  $\partial S_0$

times, generating at most  $\partial S_0$  additional clauses.

If  $X$  occurs in  $t$ , let the *excess* of  $X$  in  $t$  be the number of occurrences of  $X$  in  $t$  minus one, and  $\epsilon t$  be the sum of all excesses of variables in  $t$ . Note that  $\epsilon t = 0$  iff  $t$  is linear. Define  $\epsilon C$  as  $\epsilon t$  where  $C$  has head  $P(t)$ , 0 otherwise, and  $\epsilon S = \sum_{C \in S} \epsilon C$ . Then  $\epsilon S$  decreases strictly with (2). Moreover, in any rewrite from  $S_0$  to  $S$ ,  $\epsilon S$  is bounded by  $\max_{C \in S_0} \epsilon C$  times the number  $\#S$  of clauses in  $S$ . The former is bounded by  $\max_{C \in S_0} \epsilon C$ , since no rule creates any new clause with a higher excess, and the latter is bounded by  $\#S_0 + \partial S_0$ . So (2) can only be applied  $\max_{C \in S_0} \epsilon C \times (\#S_0 + \partial S_0)$  times. So  $\rightsquigarrow$  terminates in polynomially many steps. Any normal form  $S_*$  of  $S_0$  for  $\rightsquigarrow$  is clearly in  ${}_b\mathcal{H}_1$ . That  $S_*$  implies  $S_0$  is because the left-hand side of each  $\rightsquigarrow$  rule is implied by the right-hand side: in (1) the left-hand side is a resolvent (on  $Q$ ) of the right-hand clauses, in (2) it is a factor.  $\square$

Although  $\rightsquigarrow$  terminates in polynomially many steps, it need not terminate in polynomial *time*: starting from a clause with  $k$  variables in the head, each occurring  $q + 1$  times, rule (2) ends up producing a clause whose body contains  $q^k$  instances of  $\mathcal{B}$ . On sets of clauses with linear heads, e.g.,  $\mathcal{H}_1$  clause sets, (2) never applies, so  $\rightsquigarrow$  does terminate in polynomial time.

For each satisfiable set  $S$  of Horn clauses, and each predicate symbol  $P$ , let  $L_P(S)$  be the set of ground terms  $t$  such that  $P(t)$  is in the least Herbrand model of  $S$ .  $L_P(S)$  is the *language* recognized at *state*  $P$ . In the particular case that  $S$  consists only of *automaton clauses*  $P(f(X_1, \dots, X_n)) \leftarrow P_1(X_1), \dots, P_n(X_n)$  ( $X_1, \dots, X_n$  pairwise distinct), this coincides with the usual definition of the set of terms recognized at  $P$ ; such clauses are just tree automaton transitions from  $P_1, \dots, P_n$  to  $P$ . Accordingly, we call a set of automaton clauses a (tree) *automaton*. This connection between tree automata and Horn clauses was pioneered by Fröhlich et al. [3]; there,  $L_P(S)$  is called the *success set* for  $P$ . The point is that least Herbrand models naturally generalize  $L_P(S)$  to sets  $S$  that are not just tree automata.

By Proposition 3, for each predicate  $P$  in  $S_0$ ,  $L_P(S_*) \supseteq L_P(S_0)$ . I.e.,  $S_*$  provides *upper approximants* for success sets (languages) defined by  $S_0$ — $L_P(S_*)$  is a *type* of  $P$  in  $S_0$ . When  $S_0$  is a set of  $\mathcal{H}_1$  clauses, Proposition 3 can be refined:

**Proposition 4** *If  $S_0$  is a set of  $\mathcal{H}_1$  clauses, and  $S_0 \rightsquigarrow^* S_*$ , then  $S_0$  and  $S_*$  are equivalent up to auxiliary predicates, that is, either both are unsatisfiable, or both are satisfiable and  $L_P(S_0) = L_P(S_*)$  for every  $P$  occurring in  $S_0$ .*

Proposition 4 is subsumed by Nielson et al. [6, Proposition 2] (whose proof was omitted). There, a linear time complexity is claimed. This must be taken with a grain of salt. On Turing machines, the best we can claim is that  ${}_b\mathcal{H}_1$  and  $\mathcal{H}_1$  have equivalent expressive power, up to polynomial time reductions—since  $\rightsquigarrow$  terminates in polynomial time in this case.

**PROOF.** Starting from  $\mathcal{H}_1$  clauses,  $\rightsquigarrow$  only produces  $\mathcal{H}_1$  clauses, and (2) never

applies. We claim that if  $S$  is a set of  $\mathcal{H}_1$  clauses and  $S \rightsquigarrow S'$  (by (1)), then  $S$  logically implies  $S'$  up to auxiliary predicates, i.e., every (Herbrand) model  $I$  of  $S$  induces a (Herbrand) model  $I'$  of  $S'$  whose restriction to the predicates of  $S$  is  $I$ . This will then prove the proposition.

Let  $P(\mathcal{C}[t]) \Leftarrow \mathcal{B}$  be the clause rewritten by (1) in  $S$ , generating the fresh predicate  $Q$ . Build  $I'$  by extending  $I$  with all ground atoms  $Q(t\sigma)$ , where  $\sigma$  is any grounding substitution such that  $\mathcal{B}\sigma$  holds in  $I$ . We claim that  $P(\mathcal{C}[Z]) \Leftarrow \mathcal{B}, Q(Z)$  is valid in  $I'$ . Otherwise, there would be a grounding substitution  $\sigma'$  such that  $P(\mathcal{C}[Z]\sigma')$  is false in  $I'$  but all atoms in  $\mathcal{B}\sigma'$  as well as  $Q(u)$  are true in  $I'$ , where  $u = \sigma'(Z)$ . By definition of  $I'$ ,  $u$  is of the form  $t\sigma$ , where  $\mathcal{B}\sigma$  holds in  $I$ . We now merge  $\sigma$  and  $\sigma'$  as follows. By Definition 2 (b), no variable in  $t$  is connected to any variable in  $\mathcal{C}[\ ]$ : partition variables into  $\mathcal{V}_1$ , the set of variables connected (in  $\mathcal{B}$ ) to some variable occurring in  $t$ , and its complement  $\mathcal{V}_2$ . Let  $\sigma''$  map each  $X \in \mathcal{V}_1$  to  $\sigma(X)$ , and each  $X \in \mathcal{V}_2$  to  $\sigma'(X)$ . In particular  $\mathcal{C}[Z]\sigma'' = \mathcal{C}[Z]\sigma'$  since no variable occurring in  $\mathcal{C}[Z]$  is connected to any variable of  $t$ . So  $P(\mathcal{C}[Z]\sigma'')$  is false in  $I'$ . Since  $Z\sigma'' = Z\sigma' = u = t\sigma = t\sigma''$ ,  $P(\mathcal{C}[t]\sigma'')$  is false in  $I'$ . Also, all atoms of  $\mathcal{B}\sigma''$  are either of the form  $A\sigma$  or  $A\sigma'$  with  $A \in \mathcal{B}$ , and are therefore true in  $I'$ . So  $\sigma''$  falsifies  $P(\mathcal{C}[t]) \Leftarrow \mathcal{B}$ , contradiction.  $\square$

**Deciding  $\mathcal{H}_1$  by Resolution.** Ordered resolution with selection is a complete refinement of resolution, parameterized by a stable strict reduction ordering  $\succ$  on atoms and a *selection function* mapping each clause to a subset of its negative literals [1]. In the case of Horn clauses, ordered resolution with selection can be stated thus: from the *main* premise  $A \Leftarrow \mathcal{B}, A_1, \dots, A_m$  (where  $A_1, \dots, A_m, m \geq 1$ , is the set of selected atoms if any atom is selected at all, or  $m = 1$  and  $A_1$  is  $\succ$ -maximal in the whole clause if no atom is selected), and the  $m$  *side* premises  $A'_i \Leftarrow \mathcal{B}'_i, 1 \leq i \leq m$  (where no atom is selected and  $A'_i$  is  $\succ$ -maximal in each), infer the *resolvent*  $A\sigma \Leftarrow \mathcal{B}\sigma, \mathcal{B}'_1\sigma, \dots, \mathcal{B}'_m\sigma$  (where  $\sigma$  is the simultaneous most general unifier of  $A_1$  with  $A'_1, \dots, A_m$  with  $A'_m$ ). The resolvent is added to the current set of clauses. This is complete in the sense that  $S$  is unsatisfiable iff the empty clause  $\perp$  can be deduced by finitely many instances of this rule from  $S$ . Completeness is retained also in the presence of redundancy elimination rules [1].

Call an  $\epsilon$ -*block* any finite set of atoms of the form  $P_1(X), \dots, P_m(X)$  (with the same  $X$ , and  $m \geq 0$ ); it is *non-empty* iff  $m \geq 1$ . We shall abbreviate  $\epsilon$ -blocks  $B(X)$  to make the variable  $X$  explicit. We say that  $B(X)$  is a *block of* the clause  $A \Leftarrow \mathcal{B}, B(X)$  iff  $X$  occurs neither in  $A$  nor in  $\mathcal{B}$ . A *deep* atom is any atom of the form  $P(f(\dots))$ , i.e., not an atom of the form  $P(X)$  or  $\perp$ .

We shall use an additional rule, which is a variant of the *splitting without backtracking* rule of Riazanov and Voronkov [7], namely the  $\epsilon$ -*splitting* rule of Goubault-Larrecq et al. [4]: for each non-empty  $\epsilon$ -block  $B(X)$ , create a fresh nullary predicate symbol  $q_B$ ; now, if  $B(X)$  is a non-empty block of  $A \Leftarrow \mathcal{B}, B(X)$ , then replace the latter by the two clauses  $A \Leftarrow \mathcal{B}, q_B$  and  $q_B \Leftarrow B(X)$ . (Intuitively, the latter *defines*  $q_B$  to hold whenever the intersection of the languages of predicates in  $B$  is non-empty, and will be called a *defining clause*. The former allows one to conclude

$A$  from  $\mathcal{B}$ , as soon as the *splitting atom*  $q_B$  holds.) This replacement rule can be applied anytime without breaking completeness, provided  $A$  or  $\mathcal{B}$  contains at least one atom of the form  $P(t)$  (for any  $t$ ), and the ordering  $\succ$  is extended so that  $P(t) \succ q_B$  for every unary predicate symbol  $P$ , every term  $t$ , and every splitting atom  $q_B$ . This is an easy consequence of Bachmair and Ganzinger [1, Section 4.2.2]’s standard redundancy criterion.

To decide  ${}_b\mathcal{H}_1$ , let  $P(s) \succ Q(t)$  iff  $s$  is a proper superterm of  $t$ , regardless of  $P$  and  $Q$ . Define the selection function as follows: if the clause body contains some splitting atom  $q_B$ , select one; otherwise, if it contains a deep atom, select one; otherwise, select all atoms in the body if the head is not deep; else none. For short, call *resolution* this special brand of ordered resolution with selection.

We now show that, starting from a set of  ${}_b\mathcal{H}_1$  clauses, all clauses produced by resolution are of a special form, provided  $\epsilon$ -splitting is used *eagerly* (i.e., as soon as possible). The latter is not required to decide  ${}_b\mathcal{H}_1$ , only to reach optimal complexity bounds. Write  $\overline{X}^k$  for the sequence of pairwise distinct variables  $X_1, \dots, X_k$ .

$$\begin{array}{c}
\frac{P(f(\overline{X}^k)) \Leftarrow B_1(X_1), \dots, B_k(X_k) \quad H \Leftarrow P(f(t_1, \dots, t_k)), \mathcal{B}}{H \Leftarrow \mathcal{B}, B_1(t_1), \dots, B_k(t_k)} \quad (3) \\
\frac{\overbrace{P_j(f(\overline{X}^k)) \Leftarrow B_{j1}(X_1), \dots, B_{jk}(X_k)}^{1 \leq j \leq \ell, \ell \geq 1} \quad \overbrace{P_j(X)}^{\ell+1 \leq j \leq m} \quad H \Leftarrow P_1(X), \dots, P_m(X)}{H \Leftarrow B_{11}(X_1), \dots, B_{\ell 1}(X_1), \dots, B_{1k}(X_k), \dots, B_{\ell k}(X_k)} \quad (4) \\
\text{where } H = \perp \text{ or } H \text{ is a splitting literal } q \\
\frac{\overbrace{P_j(f(\overline{X}^k)) \Leftarrow B_{j1}(X_1), \dots, B_{jk}(X_k)}^{1 \leq j \leq \ell, \ell \geq 1} \quad \overbrace{P_j(X)}^{\ell+1 \leq j \leq m} \quad P(X) \Leftarrow P_1(X), \dots, P_m(X)}{P(f(\overline{X}^k)) \Leftarrow B_{11}(X_1), \dots, B_{\ell 1}(X_1), \dots, B_{1k}(X_k), \dots, B_{\ell k}(X_k)} \quad (5) \\
\frac{q \quad H \Leftarrow \mathcal{B}, q}{H \Leftarrow \mathcal{B}} \quad (6) \quad \frac{P_1(X) \dots P_m(X) \quad H \Leftarrow \mathcal{B}, P_1(t_1), \dots, P_m(t_m)}{H \Leftarrow \mathcal{B}} \quad (7)
\end{array}$$

Fig. 1. Specializing resolution to  ${}_b\mathcal{H}_1^{a,*}$  clauses

**Proposition 5** *Let  $a$  be an upper bound on arities of function symbols. Call  ${}_b\mathcal{H}_1^{a,*}$  the class of non- $\epsilon$ -splittable clauses of the form  $H \Leftarrow P_1(t_1), \dots, P_n(t_n), q_{B_1}, \dots, q_{B_m}$ , where  $B_1, \dots, B_m$  are non-empty  $\epsilon$ -blocks,  $m \leq a$ , and  $H$  is  $\perp$ , a splitting symbol  $q_B$ , or an atom  $P(X)$  or  $P(f(\overline{X}^k))$  (the latter two forms being as in Definition 1). Applying resolution to  ${}_b\mathcal{H}_1^{a,*}$  yields clauses that are or that  $\epsilon$ -split into clauses of  ${}_b\mathcal{H}_1^{a,*}$ . All resolution steps are of one of the forms shown in Figure 1.*

**PROOF.** Because of selection, the only clauses that can be used as side premises, and which cannot be  $\epsilon$ -split further, are unit clauses  $q_B$ , or so-called *universal clauses*  $P(X)$ , or so-called *alternating automaton clauses* of the form

$$P(f(X_1, \dots, X_k)) \Leftarrow B_1(X_1), \dots, B_k(X_k) \quad (8)$$

where  $B_1(X_1), \dots, B_k(X_k)$  are possibly empty  $\epsilon$ -blocks. (When each  $B_i(X_i)$  contains exactly one atom, these are just automaton clauses. General  $\epsilon$ -blocks encode intersection of languages inside transitions, whence the *alternating* qualifier.)

Assume some atom  $q_{B_i}$  is selected in the main premise. We can only resolve it with a unit clause  $q_{B_i}$  (if any), resulting in an  ${}_b\mathcal{H}_1^{a,*}$  clause by Rule (6). Otherwise, if the main premise is of the form  $H \Leftarrow \mathcal{B}, P(t)$ , where  $P(t)$  is deep and selected, say  $t = f(t_1, \dots, t_k)$ , then we may resolve it either with a universal clause  $P(X)$  (Rule (7)) or with a clause (8) (Rule (3)). In both cases, we get a clause  $H \Leftarrow \mathcal{B}$  or  $H \Leftarrow \mathcal{B}, B_1(t_1), \dots, B_k(t_k)$ , of the right form except that it may  $\epsilon$ -split. The largest number of blocks of the resolvent is reached when  $t$  is of the form  $f(\overline{X}^k)$ ,  $X_1, \dots, X_k$  are distinct variables occurring only in atoms of the form  $P_{ij}(X_i)$  in the body, thus creating  $k \leq a$  splitting atoms  $q_{B'_i}$ ,  $1 \leq i \leq k$  in one of the splittees, plus  $k$  defining clauses, so that the constraint  $m \leq a$  is satisfied.

In the remaining cases, the main premise is of the form  $H \Leftarrow \mathcal{B}$ , and we have two subcases. Either  $H$  is deep and no atom of  $\mathcal{B}$  is selected: then, since  $H \Leftarrow \mathcal{B}$  cannot  $\epsilon$ -split, all atoms of  $\mathcal{B}$  are of the form  $P(X)$  with  $X$  occurring in  $H$ , so none can be maximal: contradiction. Or  $H$  is not deep and all atoms of  $\mathcal{B}$ , none of which are deep or of the form  $q_B$ , are selected. Moreover,  $\mathcal{B}$  is not empty. So the main premise is of the form  $H \Leftarrow B_1(X_1), \dots, B_n(X_n)$ , where  $H$  is of one of the forms  $\perp$ ,  $q_B$ , or  $Q(X)$ . By assumption this is not  $\epsilon$ -splittable, so  $n = 1$  (and  $X_n = X$  if the head is  $Q(X)$ ). That is, the main premise is of the form  $H \Leftarrow B_1(X)$  where  $B_1(X)$  is a non-empty block, and  $H$  is  $\perp$ ,  $q_B$  or  $Q(X)$  (with the same  $X$ ). The side premises are universal clauses of the form  $P_j(X)$  with  $P \in B_1$ , or alternating automaton clauses  $P_j(f(\overline{X}^k)) \Leftarrow B_{j1}(X_1), \dots, B_{jk}(X_k)$  with possibly different  $P_j \in B_1$ , but with the same  $f$ . Index them so that  $1 \leq j \leq \ell$  in the latter and  $\ell + 1 \leq j \leq m$  in the former: this is Rule (4) or Rule (5) in case  $\ell \geq 1$ , Rule (7) if  $\ell = 0$ .  $\square$

This allows us to restate Nielson et al. [6]’s main theorem, in a form that makes the role of the maximal arity  $a$  more apparent:

**Theorem 6** *Let  ${}_b\mathcal{H}_1^a$ , resp.  $\mathcal{H}_1^a$  be the restriction of  ${}_b\mathcal{H}_1$ , resp.  $\mathcal{H}_1$  to clause sets with function symbols of arity at most  $a$ . Satisfiability of  ${}_b\mathcal{H}_1$ , resp.  $\mathcal{H}_1$  clause sets, as well as  ${}_b\mathcal{H}_1^a$ , resp.  $\mathcal{H}_1^a$  clause sets for any fixed  $a \geq 1$ , is DEXPTIME-complete.*

**PROOF.** We may assume without loss of generality that all clauses in the initial set are  $\epsilon$ -split. Let  $n_p$  be the number of unary predicate symbols,  $n_f$  be the number of function symbols. So there are at most  $2^{n_p}$  splitting atoms  $q_B$ . Since  $m \leq a$  in  ${}_b\mathcal{H}_1^{a,*}$  clauses, there are at most  $\sum_{m=0}^a \binom{2^{n_p}}{m} \leq \sum_{m=0}^a \frac{2^{n_p m}}{m!} \leq \sum_{m=0}^a \frac{2^{n_p a}}{m!} < e 2^{n_p a}$  subsets  $q_{B_1}, \dots, q_{B_m}$ . Look at the rules in Figure 1. They produce either alternating automaton clauses (Rule (5)), of which there are at most  $\underbrace{n_p n_f}_{\text{head}} \underbrace{2^{n_p a}}_{\text{body}}$ ; or clauses con-

taining only subterms of terms occurring in the right premise (this is obvious for all rules except Rule (4); for the latter, remember that the conclusion  $\epsilon$ -splits as a collection of one-variable clauses, and we may rename this variable as the variable  $X$  of the right premise). By induction on the length of the derivation, all derived clauses that are not alternating automaton clauses only contain subterms of the initial set of clauses. Let  $N$  be the number of such subterms, so we can derive at most  $(1 + 2^{n_p} + n_p + n_p n_f) \underbrace{2^{n_p N}}_{P_i(t_i)} \underbrace{e 2^{n_p a}}_{q_{B_j}}$  non-alternating automata clauses. Hence resolution with eager  $\epsilon$ -splitting only generates exponentially many clauses.

Conversely, emptiness of alternating two-way (word) automata  $S$  is DEXPTIME-complete [2, Corollary 4];  $S$  is an  ${}_b\mathcal{H}_1^1$  clause set, and we can test whether  $L_P(S) = \emptyset$  by adding  $\perp \Leftarrow P(X)$  and checking for satisfiability.  $\square$

In contrast to Theorem 6, if  $a = 0$ ,  ${}_b\mathcal{H}_1^a$  and  $\mathcal{H}_1^a$  can be decided in polynomial time, by instantiating all clauses over the finite Herbrand base.

An easy application is *Tison's Theorem* [5, Theorem 6]: given a regular tree language  $L_P(S)$ , it is decidable in exponential time whether  $L_P(S)$  (where  $S$  is a set of automaton clauses) contains an instance of some given first-order term  $s$ —even when  $s$  is *not* linear. Indeed, this is equivalent to checking whether  $S$  plus the clause  $\perp \Leftarrow P(s)$  is unsatisfiable. The results above establish that this is still decidable in exponential time, and no more, when  $S$  is presented as an alternating automaton, or even as a set of  ${}_b\mathcal{H}_1$  or  $\mathcal{H}_1$  clauses. Also, our technique does not require determinizing any automaton, which makes it potentially more efficient than the algorithm of Middeldorp [5]. Let us push the argument further:

**Theorem 7** *It is decidable in exponential time whether, given  $n$  terms  $t_1, \dots, t_n$ , possibly sharing free variables, and  $n$  tree languages  $L_{P_1}(S_1), \dots, L_{P_n}(S_n)$ , whether there is a substitution  $\sigma$  such that  $t_i\sigma$  is ground and in  $L_{P_i}(S_i)$  for all  $i$ ,  $1 \leq i \leq n$ . This holds whether tree languages are presented by tree automata, alternating automata, or satisfiable  ${}_b\mathcal{H}_1$  or  $\mathcal{H}_1$  clause sets.*

Indeed, without loss of generality,  $S_1, \dots, S_n$  are built using disjoint sets of predicates. Let  $S$  be  $\bigcup_{i=1}^n S_i$ .  $S$  plus the clause  $\perp \Leftarrow P_1(t_1), \dots, P_n(t_n)$  is unsatisfiable iff there is a grounding substitution  $\sigma$  such that  $t_1\sigma \in L_{P_1}(S), \dots, t_n\sigma \in L_{P_n}(S)$ .

**Compiling  ${}_b\mathcal{H}_1$  to Alternating Automata, and to Automata.** Call sets of universal clauses and alternating automaton clauses *alternating automata*. This coincides with usual alternating tree automata, once final states are chosen, and provided universal clauses are replaced by the usual clauses defining universal languages.

Starting from an  ${}_b\mathcal{H}_1$  clause set  $S_0$ , resolution with  $\epsilon$ -splitting ends with a so-called *saturated* set  $S$  of clauses. It is well-known that (at least without splitting) we can extract a model from the latter by taking all clauses from which no atom is selected. Let  $S^-$  be the set of universal and alternating automaton clauses in  $S$  (we do not keep unit clauses  $q_B$ ). We can in fact prove directly that  $S^-$  and  $S$  have the

same least Herbrand model. Take any ground atom  $P(t)$ .  $P(t)$  is in the least Herbrand model of  $S$  iff  $S$  plus  $\perp \Leftarrow P(t)$  is unsatisfiable. But since  $S$  is saturated, any derivation of  $\perp$  from  $S$  plus  $\perp \Leftarrow P(t)$  by resolution and eager  $\epsilon$ -splitting must first resolve  $\perp \Leftarrow P(t)$  with some clause with no selected literal from  $S$ , i.e., with some clause from  $S^-$ . The resolvent must then be of the form  $\perp \Leftarrow P_1(t_1), \dots, P_n(t_n)$  (a *negative ground clause*). An easy induction on the length of the refutation shows that any refutation from  $S$  plus a set of negative ground clauses can only resolve between the latter and clauses from  $S^-$ . Since no clause from  $S \setminus S^-$  is ever used,  $P(t)$  is in the least Herbrand model of  $S^-$ . Conversely, the least Herbrand model of  $S^-$  is trivially included in that of  $S$ . So:

**Proposition 8** *Any satisfiable  $\mathcal{H}_1$  clause set  $S_0$  can be converted in exponential time to an equivalent alternating automaton  $S^-$  recognizing the same languages:  $L_P(S^-) = L_P(S_0)$  for every predicate  $P$ .*

Alternating automata can now be converted to (non-deterministic) tree automata, using a variant of *non-ground splitting*. Our version of this rule reads as follows. For each  $\epsilon$ -block  $B(X)$  containing at least two atoms, create a fresh predicate symbol  $[B]$ —call such predicates the *intersection predicates*. Then we may replace any clause  $H \Leftarrow \mathcal{B}, B(X)$  of which  $B(X)$  is a block of at least two atoms, by the two clauses  $H \Leftarrow \mathcal{B}, [B](X)$  and  $[B](X) \Leftarrow B(X)$ . Generalize this rule: replace any clause  $H \Leftarrow \mathcal{B}, B(X), [B_1](X), \dots, [B_n](X)$ , where  $(B, B_1, \dots, B_n)$  contains no intersection predicate and at least two distinct atoms, and  $X$  does not occur in  $H \Leftarrow \mathcal{B}$ , by the two clauses  $H \Leftarrow \mathcal{B}, [B, B_1, \dots, B_n](X)$  and  $[B, B_1, \dots, B_n](X) \Leftarrow B(X), B_1(X), \dots, B_n(X)$ . Completeness is preserved if we apply this non-ground splitting rule at any time, using standard redundancy arguments [1, Section 4.2.2], provided that  $\succ$  is extended so that  $P(s) \succ [B](t)$  for any non-intersection predicate  $P$ , and  $[B](s) \succ [B'](t)$  whenever  $B(s)$  properly subsumes  $B'(t)$ . The arguments of Theorem 6 apply to resolution with eager  $\epsilon$ -splitting and this form of non-ground splitting. Given a saturated set  $S$ , the set  $S^-$  then consists of universal clauses  $P(X)$  and clauses of the form  $P(f(\overline{X}^k)) \Leftarrow P_{i_1}(X_{i_1}), \dots, P_{i_\ell}(X_{i_\ell})$  ( $1 \leq i_1 < \dots < i_\ell \leq k$ ), which can clearly be transformed into equivalent automaton clauses in polynomial time. So:

**Proposition 9** *Any satisfiable  $\mathcal{H}_1$  clause set can be converted in exponential time to an equivalent tree automaton recognizing the same languages. In particular, all languages  $L_P(S)$ ,  $S$  a set of  $\mathcal{H}_1$ , resp.  $\mathcal{H}_1$  clauses, are effectively regular.*

**The  $\mathcal{H}_2^{i,a}$  subclass.** For any  $i, a \in \mathbb{N}$ , let  $\mathcal{H}_2^{i,a}$  consist of those  $\mathcal{H}_1^a$  clauses such that any variable occurring in the head occurs at most once in the body, and any other variable occurs at most  $i$  times in the body. These invariants are preserved by resolution and eager  $\epsilon$ -splitting. There are now at most  $n_p n_f (n_p + 1)^a$  alternating automaton  $\mathcal{H}_2^{i,a}$  clauses (and they are all just automaton clauses). Assume without loss of generality that  $i \leq n_p$ . Since we only need splitting atoms  $q_B$  with  $|B| \leq i$ , we need at most  $\sum_{m=1}^i \binom{n_p}{m} \leq en_p^i$  of them. In particular, at most  $en_p^i$  defining

clauses are generated, each of size  $O(n_p^i)$ . Finally, all cases of resolution of Figure 1 now produce smaller and smaller clauses (for rule (3), this is because  $B_1, \dots, B_k$  contain at most one atom each by definition of  ${}_b\mathcal{H}_2^{i,a}$ ), except for Rule (5) which, as we have seen, generates an alternating automaton clause, and Rule (4), which only generates clauses  $H \Leftarrow B_1(X)$  with  $H$  of the form  $\perp$  or  $q_B$  and  $|B_1| \leq i$ , hence can only generate at most  $2^i$  consecutive clauses of the same size;  $\epsilon$ -splitting produces defining clauses, which are only polynomially many anyway, plus a smaller clause (counting the size of  $q_B$  as being equal to that of  $B(X)$ ). Our version of non-ground splitting never applies on clauses that are already  $\epsilon$ -split. So:

**Proposition 10** *Let  $i, a \in \mathbb{N}$  be fixed. Satisfiability and conversion of  ${}_b\mathcal{H}_2^{i,a}$  clause sets to tree automata can be done in polynomial time.*

This is roughly Theorem 2 of Nielson et al. [6]. Fixing  $i$  and  $a$  is important: the complexity of  ${}_b\mathcal{H}_2^{i,a}$  is *exponential*, not polynomial, in  $i$  and  $a$ . In fact, satisfiability of  ${}_b\mathcal{H}_2^a = \bigcup_{i \in \mathbb{N}} {}_b\mathcal{H}_2^{i,a}$  clause sets is DEXPTIME-complete for any  $a \geq 2$ , since it easily encodes the emptiness problem for intersections of tree automata [8], using a goal  $\perp \Leftarrow P_1(X), \dots, P_i(X)$  to test whether  $L_{P_1}(S) \cap \dots \cap L_{P_i}(S) = \emptyset$ .

Clearly,  ${}_b\mathcal{H}_2 = \bigcup_{a \geq 0} {}_b\mathcal{H}_2^a$  is a subclass of  $\mathcal{H}_2$ , the subclass of  $\mathcal{H}_1$  where (\*) every variable in the head occurs at most once in the body [6]. Conversely, every  $\mathcal{H}_2$  clause set converts in polynomial time, using (1) to an  ${}_b\mathcal{H}_2$  clause set, by Proposition 4 and noticing that rule (1) preserves (\*).  $\mathcal{H}_2$  and  ${}_b\mathcal{H}_2$  are therefore equivalent up to polynomial time reductions.

**Undecidability.** May we add equality tests to  ${}_b\mathcal{H}_1$  clause sets, much as with tree automata with equality tests between brothers? In principle, this is easy: drop the requirement that  $X_1, \dots, X_k$  be distinct in Definition 1, to obtain the new class  ${}_b\mathcal{H}_1^-$ . E.g.,  $P(f(X, X)) \Leftarrow Q(X)$  states that the terms recognized at  $P$  are of the form  $f(s, t)$  with  $s = t$ , and  $t$  recognized at  $Q$ .

**Theorem 11** *It is undecidable whether sets of  ${}_b\mathcal{H}_1^-$  clauses are satisfiable.*

**PROOF.** Encode Post's correspondence problem on two letters  $a, b$ . Given any word  $w$ , and any term  $t$ , define the term  $w(t)$  by:  $\epsilon(t) = t$ ,  $aw(t) = a(w(t))$ ,  $bw(t) = b(w(t))$ . Write the following clauses. First,  $Q_c(c); Q(a(X)) \Leftarrow Q_c(X); Q(b(X)) \Leftarrow Q_c(X); Q(a(X)) \Leftarrow Q(X); Q(b(X)) \Leftarrow Q(X)$ ; so that  $Q$  recognizes all terms  $w(c)$ ,  $w \in \{a, b\}^+$ . Then the only non- ${}_b\mathcal{H}_1$  clause  $P(f(X, X)) \Leftarrow Q(X)$ ;  $P$  recognizes all pairs of equal non-empty words. Then,  $P'(X) \Leftarrow P(X)$ , and for each Post pair  $(u, v)$ ,  $P'(f(X, Y)) \Leftarrow P'(f(u(X), v(Y)))$ . Post's correspondence problem has a solution iff these clauses, plus  $\perp \Leftarrow P'(f(c, c))$ , form an unsatisfiable set. Note that we need only one non- ${}_b\mathcal{H}_1$  clause, and all others are in  ${}_b\mathcal{H}_2^{0,2}$ .  $\square$

**Note added to the final version.** At the same time that this paper was accepted, H. Seidl and I discovered that some of the results of this paper were already present

in [9]. There,  $\text{b}\mathcal{H}_1$  clause sets are called “monadic Horn theories where all positive literals are linear and shallow”. That  $\text{b}\mathcal{H}_1$  can be decided by sort resolution is the topic of [9, Lemma 4]. No complexity bound is given; the procedure of op.cit. runs in double exponential time, and is therefore not optimal. That the problem mentioned in Theorem 7 is decidable was also proved, in [9, Lemma 3], again without complexity bound. Finally, [9] gives another proof that  $\text{b}\mathcal{H}_1^-$  is undecidable (end of Section 3), by reduction from unifiability in arbitrary sort theories.

**Acknowledgments.** Thanks to H. Comon-Lundh, who urged me to publish these results, to H. Seidl for spotting a bug in a previous version of Proposition 4, to F. Jacquemard, who carefully reread a first version of this paper, and to the anonymous referees for pointing out some important bugs and inaccuracies.

## References

- [1] L. Bachmair and H. Ganzinger. Resolution theorem proving. In J. A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume I, chapter 2, pages 19–99. North-Holland, 2001.
- [2] W. Charatonik, A. Podelski, and J.-M. Talbot. Paths vs. trees in set-based program analysis. In *Proc. 27th Ann. ACM Symp. Principles of Programming Languages*, pages 330–338. ACM Press, 2000.
- [3] T. Frühwirth, E. Shapiro, M. Y. Vardi, and E. Yardeni. Logic programs as types for logic programs. In *Proc. 6th Symp. Logic in Computer Science*, pages 300–309. IEEE Computer Society Press, 1991.
- [4] J. Goubault-Larrecq, M. Roger, and K. N. Verma. Abstraction and resolution modulo AC: How to verify Diffie-Hellman-like protocols automatically. *J. Logic and Algebraic Programming*, 2004. To appear.
- [5] A. Middeldorp. Approximating dependency graphs using tree automata techniques. In F. Massacci and R. Goré, editors, *Proc. 1st Intl. Joint Conf. Automated Reasoning*, pages 593–610. Springer-Verlag LNAI 2083, 2001.
- [6] F. Nielson, H. R. Nielson, and H. Seidl. Normalizable Horn clauses, strongly recognizable relations and Spi. In *Proc. 9th Static Analysis Symposium*, pages 20–35. Springer Verlag LNCS 2477, 2002.
- [7] A. Riazanov and A. Voronkov. Splitting without backtracking. In B. Nebel, editor, *Proc. 17th Intl. Joint Conf. Artificial Intelligence*, volume 1, pages 611–617. Morgan Kaufmann, Aug. 2001.
- [8] H. Seidl. Haskell overloading is DEXPTIME-complete. *Inf. Proc. Letters*, 52(2):57–60, 1994.
- [9] C. Weidenbach. Towards an automatic analysis of security protocols in first-order logic. In *Proc. 16th Intl. Conf. Automated Deduction*, pages 314–328. Springer-Verlag LNAI 1632, 1999.