Complexity Analysis Based on Ordered Resolution*

David Basin Harald Ganzinger
Max-Planck-Institut für Informatik,
Im Stadtwald, D-66123 Saarbrücken, Germany
{basin,hg}@mpi-sb.mpg.de

Abstract

We define order locality *to be a property of clauses relative to a term ordering. This property is a kind of generalization of the subformula property for proofs where terms arising in proofs are bounded, under the given ordering, by terms appearing in the goal clause. We show that when a clause set is order local, then the complexity of its ground entailment problem is a function of its structure (e.g., full versus Horn clauses), and the ordering used. We prove that, in many cases, order locality is equivalent to a clause set being saturated under ordered resolution. This provides a means of using standard resolution theorem provers for testing order locality and transforming non-local clause sets into local ones. We have used the Saturate system to automatically establish complexity bounds for a number of nontrivial entailment problems relative to complexity classes which include polynomial and exponential time and co-NP.*

1. Introduction

Many algorithms and decision problems can be reduced to ground entailment problems: determining if a ground formula is entailed by a given theory. For example, congruence closure is the problem of whether $s = t$ follows from ground equations $s_1 = t_1, \ldots, s_n = t_n$, in a theory that axiomatizes equality to be reflexive, symmetric, transitive, and a congruence. These axioms (see Section 8.) are Horn clauses and hence congruence closure is the decision problem of whether a ground Horn clause, $s_1 = t_1, \ldots, s_n = t_n \rightarrow s = t$, is entailed by a Horn clause theory.

Our research is motivated by McAllester [7, 12] who gives procedures for recognizing sets of Horn clauses for which ground entailment is polynomial time decidable. He tests if Horn clauses have a property, which he calls locality, which roughly corresponds to a kind of subformula property for Horn clause proofs; when a clause set has this property, then there are a bounded number of terms (dependent only on the query) which need to appear in proofs and this is used to solve ground queries in polynomial time.

Our contributions are threefold. First, we generalize locality by introducing *order locality*, which is a property of clauses, parameterized by term orderings. This is a strict generalization both as it is parameterized by orderings (McAllester's definition is relative to a fixed ordering, the subterm ordering), and as it is defined for full, as opposed to Horn clauses. We show how the complexity of the ground entailment problem for an order local clause set is determined by the ordering and the structure of the clauses. For example, congruence closure is local under a *polynomially bounded ordering*, which is one where every ground term has polynomially many smaller terms, and this implies polynomial time decidability. We show that tighter bounds are possible too and how other classes of orderings capture other complexity classes (co-NP, exponential time, etc.).

Second, we prove that order locality is closely related and in many cases is equivalent to a clause set being *saturated up to redundancy under ordered resolution* (defined in Section 3.). This provides both an alternative characterization of when a clause set is local and yields a procedure whereby standard ordered-resolution based theorem provers may be used to test locality. We have used the Saturate system [6] to give machine checked proofs of the complexity of a number of nontrivial entailment problems. This includes automatically verifying that congruence closure, the theory of tree-embeddings, and the theory of nonstrict partial orderings are polynomial time decidable, that propositional logic is in both co-NP and exponential time, and that the first-order theory of total orderings is in co-NP. Note that saturation can not only detect locality in many cases (order locality and saturation are both undecidable), but for many non-local clause sets, saturation can still terminate and yield a local presentation of the same theory; this is analogous to the rôle played by Knuth/Bendix completion in the equational

*This research was supported in part by the ESPRIT Basic Research Working Groups 6028 (Construction of Computational Logics) and 6112 (COMPASS). We thank David McAllester for helpful comments on a draft of this paper.

setting.

Our third contribution is to show that order locality provides not only a sufficient criterion for establishing the complexity of entailment problems, but in many cases it is *necessary* too. We show that when the ground entailment problem for a Horn clause set N belongs to one of the standard deterministic time complexity classes (polynomial time, exponential time, etc.) then N is equivalent to a clause set that is local under a corresponding ordering (e.g., polynomial, exponential, etc.). This indicates that locality and saturation are natural and powerful concepts for capturing complexity classes.

Overall, the results of this paper are a first step towards computer-assisted complexity analysis for theories and logic programs. The rôle of the user is to provide a theory presentation together with a well-founded ordering, including an analysis of how many smaller terms each ground term has. The automatic part is saturation up to redundancy of the presentation under ordered resolution. Redundancy analysis is essentially a process that aims at finding transformations for proofs that are too complex. The more intelligence one builds into this analysis, the less explicit the presentation and ordering need to be.

2. Preliminaries

We shall use standard terminology for clauses and clause-based proofs [2]. We use sequent notation $\Gamma \rightarrow \Delta$ for clauses where Γ and Δ are possibly empty multisets of atoms. Given a set of clauses N, the *entailment problem* for N is to decide if $N \models C$, for ground clauses C, called *queries*. A *refutational proof* of $N \models C$ is a tree whose root is labeled by the empty clause, and whose internal nodes are labeled by results of resolution or factoring inferences from their ancestors. Leaves are ground instances of N or unit clauses $\neg L$, with L a literal in C. When N is a set of Horn clauses, a *direct proof* of a ground atom A is a tree labeled by atoms whose root is A and whose subtrees are direct proofs of atoms that constitute the body of a ground instance of N with head A. A direct proof of a ground clause $A_1, \ldots, A_n \rightarrow A$ from N is a direct proof of A from $N \cup \{A_1, \ldots, A_n\}$. A term t *occurs* in either kind of proof if t occurs in a clause or in an atom labeling the proof tree. In this paper we

McAllester's work is based on Horn clauses and he defines a set of Horn clauses N to be *local* if for every ground Horn clause C, we have $N \models C$ if and only if C is entailed by those ground instances of N in which all terms are subterms of ground terms in N and C. Locality is a desirable property of clause sets because when N is local, then $N \models C$ precisely when $N' \models C$ for a ground Horn set N' which is polynomial in the size of C. Namely, N' comprises those ground substitution instances of N where all terms in N' are subterms in N and C. The entailment problem for N' can be efficiently solved:

Lemma 1 ([5]) *For finite sets of ground Horn clauses N and ground Horn clauses C, $N \models C$ is decidable in time linear in the sum of the sizes of N and C.*

Hence, when N is local, its ground entailment problem is tractable.

Lemma 2 ([12]) *If N is a finite, local set of Horn clauses, then for all ground Horn queries C, $N \models C$ is polynomial time decidable in the size of C.*

We generalize locality by allowing arbitrary term orderings and full clauses for our clause sets N and our queries C. Let $\succ$ be a well-founded, partial ordering on terms. We say that a set N of clauses *entails a clause C bounded by* $\succ$, and write $N \models_{\preceq} C$, if and only if there is a proof of $N \models C$ from those ground instances of N in which each term is smaller or equal with respect to $\succeq$ than some term in C. A set of clauses N is *local with respect to* $\succ$ if whenever $N \models C$ for a ground clause C, then $N \models_{\preceq} C$. To distinguish this generalized notion of locality from McAllester's, we also refer to it as *order locality* in contrast to McAllester's definition which we refer to as *subterm locality*.

3. Ordered Resolution

We define the concept of saturation up to redundancy of a set of clauses by ordered resolution. The definitions in this section are with respect to any well-founded, total ordering $\succ$ on ground *atoms*. We call such an ordering *compatible* with a total ordering $\succ'$ on ground terms, if $\max(t_i) \succ' \max(s_j)$ implies that $p(t_1 \ldots t_k) \succ q(s_1 \ldots s_n)$, for any two predicate symbols p and q.

Given an atom ordering $\succ$, we will call A [strictly] maximal with respect to a multiset of atoms Γ, if for any atom B in Γ we have $B \not\succ A$ [$B \not\succeq A$]. If the ordering is defined on ground atoms, we lift it to non-ground atoms A and B by: $A \succ B$ if $A\sigma \succ B\sigma$ for every ground substitution σ.

To extend $\succ$ to clauses, we identify a positive literal A with the singleton multiset $\{A\}$, a negative literal $\neg A$ with the multiset $\{A, A\}$, and a clause with the multiset of its literals. Then, for clauses C and C', we define $C \succ C'$ if and only if $C\ (\succ_{ms})_{ms}\ C'$ in the twofold multiset extension of $\succ$. This definition basically says that a clause is greater than another clause if the maximal atom of the former is greater than the maximal atom of the latter. In case they are identical, negative occurrences are considered greater than positive ones. If the maximal atom has identical occurences in both clauses, non-maximal atoms are compared with respect to the atom ordering and polarity. Clearly, if $\succ$ is total [well-founded] on atoms, so is its extension to clauses. Given a

ground clause C and a set of clauses N, by N_C we denote the set of ground instances D of N such that $C \succ D$ in the clause ordering.

We employ a calculus for ordered resolution based on [2]. Inference steps are by *ordered resolution* or *ordered factoring*, which take the form

$$\frac{\Gamma_1 \to A, \Delta_1 \qquad B, \Gamma_2 \to \Delta_2}{\Gamma_1\sigma, \Gamma_2\sigma \to \Delta_1\sigma, \Delta_2\sigma} \qquad \frac{\Gamma \to A, B, \Delta}{\Gamma\sigma \to A\sigma, \Delta\sigma}$$

where σ is the mgu of A and B. For ordered resolution, $A\sigma$ must be strictly maximal in $\Gamma_1\sigma, \Delta_1\sigma, \Delta_2\sigma$ and maximal in $\Gamma_2\sigma$. For ordered factoring, $A\sigma$ must be strictly maximal in $\Gamma\sigma$ and maximal in $\Delta\sigma$.

We define a notion of redundancy that identifies clauses and inferences that are not needed for establishing refutational completeness of ordered resolution. It is based on the observation that ordered inferences are monotone and, hence, need only be applied to certain minimal clauses. A ground clause C (not necessarily an instance of N) is called *redundant in* N (with respect to $\succ$) if it is entailed by the instances of N which are smaller than C, i.e., if $N_C \models C$.[1]

A ground inference by ordered resolution of a clause $\Gamma_1, \Gamma_2 \to \Delta_1, \Delta_2$ from premises $C = \Gamma_1 \to A, \Delta_1$ and $D = A, \Gamma_2 \to \Delta_2$ is called *redundant in* N (with respect to $\succ$) if (i) either premise is redundant in N, or else if (ii) $N_D \models \Gamma_1, \Gamma_2 \to \Delta_1, \Delta_2$, that is, the conclusion is entailed by instances of N smaller than the second premise. Note that since $\succ$ is total on ground atoms, the second premise of an ordered resolution inference from ground clauses is the maximal clause among the two premises and the conclusion.

We call a ground inference by ordered factoring *redundant in* N if the conclusion follows from instances in N which are smaller with respect to $\succ$ than its premise. A non-ground inference is called *redundant* if all its ground instances are redundant. Finally, we call a set of clauses *saturated up to redundancy* (with respect to $\succ$), if any inference by ordered resolution or factoring from premises in N is redundant in N. As an inference is redundant whenever the conclusion is an instance of N, we can saturate N by systematically adding all conclusions of non-redundant inferences.

Given a total and well-founded ordering $\succ$ on ground *terms*, we call N *saturated up to redundancy* (by ordered resolution with respect to $\succ$) if N is saturated with respect to some total and well-founded ordering on ground atoms that is compatible with $\succ$. If the ordering $\succ$ on ground terms is *partial*, we call N *saturated up to redundancy with respect to* $\succ$ iff N is saturated with respect to every well-founded term ordering that extends $\succ$ and is total on ground terms.

If N is saturated under ordered resolution up to redundancy with respect to $\succ$, then N is inconsistent if and only if N contains the empty clause [1]. Moreover, in saturated sets N, any inference from premises both in N is redundant and redundancy of clauses and inferences is preserved by adding clauses and deleting redundant clauses that arise during theorem proving. This accounts for the following theorem, which states that entailment proofs with respect to a saturated N possess an N*-linear* form.

Theorem 1 ([2]) *Let N be saturated and let C be a clause. C follows from N iff there is a proof, by ordered resolution, of the empty clause from a (finite) set of ground instances in $N \cup \neg C$ such that for each inference at least one premise is not an instance of N.*

4. Saturation and Locality

4.1. Saturation Implies Locality

We start by showing that saturation implies locality for total term orderings.

Lemma 3 *Let $\succ$ be a total and well-founded ordering on ground terms. Moreover, let N be saturated with respect to $\succ$ and let C be a ground clause. Then $N \models C$ iff $N \models_{\preceq} C$.*

Proof. The right to left direction is immediate. Conversely, assume $N \models C$ and that N is saturated with respect to an atom ordering $\succ$ that is compatible with the given term ordering. By Theorem 1 there exists an N-linear proof of the empty clause from a finite set of ground instances in $N \cup \neg C$. By induction on the depth of clauses in the proof we show that all terms are bounded (i.e., less than or equal under $\succeq$) by terms in C.

For the base case (depth one), consider any instance $\Gamma \to A, \Delta$ of N that is a leaf in the proof and is the first premise of an ordered resolution inference. The second premise must be a unit clause $A \to$ from $\neg C$. As the inference is ordered and $\succ$ is total, $A \succ B$, for any B in Γ or Δ. Alternatively $A, \Gamma \to \Delta$ is an instance of N that is the second premise in an ordered resolution inference, and the first premise is a unit clause $\to A$ from $\neg C$. Since A is maximal, we have that $max_{t \in A}(t) \succeq max_{s \in B}(s)$ for any B in Γ, Δ. Note that ordered factoring at depth 1 is impossible since all clauses in $\neg C$ are unit.

Now consider an ordered resolution inference of depth greater than one from premises D and D'. If both D and D' are non-leaf nodes in the proof then, by the induction hypothesis, their terms are bounded by terms in C. As the conclusion D'' of the inference is smaller than the second premise, its terms are again bounded by the terms in C. If

[1] Note the difference between redundancy, i.e., $N_C \models C$, and bounded entailment, i.e., $N \models_{\preceq} C$. When using a compatible atom ordering, there are clauses, whose maximum term is the maximum term in C, which are disallowed in proofs of $N_C \models C$ but allowed in proving $N \models_{\preceq} C$. A less significant difference is that the former is based on the clause ordering and the latter on term orderings.

only one, D say, of the two premises is an inner node, the other, D' is an instance of N or a clause from $\neg C$. Applying the induction hypothesis to D, the reasoning is essentially the same as in the base case. Ordered factoring at depth > 1 does not create new atoms, and by the induction hypothesis all the old atoms are bounded by atoms in C. □

We show in Section 6. that the complexity of entailment problems depends on how many smaller terms each ground term has under $\succ$. For most total orderings, a ground term has at least exponentially many smaller terms and therefore the direct use of these orderings would lead to poor complexity results. We get better complexity bounds by working instead with partial orders, where saturation refers to all total extensions.

Theorem 2 *Let $\succ$ be a well-founded partial ordering on terms. If N is saturated up to redundancy with respect to $\succ$, then N is local with respect to $\succ$.*

Proof. Let N be saturated up to redundancy with respect to $\succ$. Then, by definition, N is saturated up to redundancy with respect to each extension $\succ'$ of $\succ$ that is well-founded and total on ground terms. We show that N is local with respect to $\succ$. Let C be a ground clause such that $N \models C$. We have to prove that $N \models_{\preceq} C$. To this end, let $\succ'$ be an ordering that (i) extends $\succ$, (ii) is well-founded, (iii) is total on ground terms, and satisfies (iv) $s \succ' t$, for any ground term t in C and any ground term s for which there is no term u in C such that $u \succeq s$. In other words, we construct $\succ'$ in a way such that all ground terms that are not smaller or equal to some term in C with respect to $\succ$ become strictly bigger with respect to $\succ'$ than any term in C. As N is saturated with respect to $\succ'$, we may apply Lemma 3 and infer that $N \models_{\preceq'} C$ and by (iv) we have $N \models_{\preceq} C$. □

4.2. Saturation is Weaker than Locality

The converse of the previous theorem is not true in general; however it does hold in restricted cases.

Theorem 3 *Suppose that we have just one predicate symbol which is, in addition, unary. Let N be a set of Horn clauses and let $\succ$ be a well-founded partial ordering on terms. If N is local with respect to $\succ$ then N is saturated up to redundancy with respect to $\succ$.*

Proof. Let p be the unary predicate symbol, and let $\succ'$ be an extension of $\succ$ that is well-founded and total on ground terms. The ordering on ground atoms, defined by $p(s) \succ' p(t)$ iff $s \succ' t$, is total, well-founded and compatible with the term ordering. Consider an inference by ordered resolution with respect to $\succ'$ from ground instances of clauses in N:

$$\frac{\Gamma_1 \to A \qquad A, \Gamma_2 \to B}{\Gamma_1, \Gamma_2 \to B}$$

A is strictly greater (in $\succ'$) than any of the atoms in the clause $C = \Gamma_1, \Gamma_2' \to B$, where $\Gamma_2' = \Gamma_2 \setminus \{A\}$. Since C is a logical consequence of N and N is local with respect to $\succ$, we have $N \models_{\preceq} C$, that is, C is implied by those instances of N in which all atoms are smaller or equal (in $\succeq$) than some atom in C. These instances of N are all smaller than A, and, in particular, smaller than $A, \Gamma_2 \to B$ with respect to $\succ'$. As C implies the conclusion of the inference we have shown that the inference is redundant. □

Factoring inferences may not be redundant if N also contains non-Horn clauses. One may generalize the Theorem to full clauses if factorization is built into the resolution inferences, that is, if one considers clauses as sets, rather than multisets, of literals. However, the main restriction in Theorem 3 concerns the signature. Suppose N is the following set of clauses.

$$\begin{aligned} p(f(x)) &\to q(x) \\ p'(f(x)) &\to q'(x) \\ p'(x) &\to p(x) \\ p(x) &\to p'(x) \\ r(f(x)) &\to p(x) \\ r'(f(x)) &\to p'(x) \end{aligned}$$

N is local with respect to the subterm ordering, but whatever ordering we choose on atoms, N will not be saturated under ordered resolution. Suppose that a is a constant and that, for instance, $p'(f(a)) \succ p(f(a))$. Then $p(f(a)) \to q'(a)$ can be obtained by ordered resolution from instances of the second and fourth clause, but cannot be obtained by any other proof that does not involve the maximal atom $p'(f(a))$ of the inference.

We observe that requiring atoms to be ordered may be too restrictive when they contain the same maximal term. Moreover, clauses that contain the maximal term both in the antecedent and in the succedent may require one to consider non-order-local proofs of potentially unbounded depth in which a path of the form $A, A_1, \ldots, A_n, B$ (a plateau) exists such that the A_i, but neither A nor B, contain the maximal term.

In the full paper [4] we consider ordered resolution with selection, and establish essentially the same results as in Theorems 2 and 3. In ordered resolution with selection among the negative literals that contain the maximal term, an arbitrary, not necessarily maximal, literal may be selected. Selection improves the situation in some, but not in all cases. In general, to eliminate "plateaus" of unbounded width requires a stronger form of redundancy that we will discuss in the following section.

5. The Horn Clause Case

For full clauses we have shown that saturation implies locality; however, only for a quite restricted class of signatures and orderings we have been able to prove the converse too (cf. Theorem 3). The purpose of this section is to show that saturation up to redundancy and order locality are in fact equivalent concepts in the case of Horn clauses provided the notion of redundancy is appropriately extended. *Throughout this section we assume that all clauses are Horn.*

5.1. Monotonicity

To simplify the technical treatment, we will generalize order locality to *monotonicity* by restricting the class of queries considered. A set of Horn clauses N is *monotone* with respect to a well-founded partial ordering $\succ$ on terms if for any ground *atom* A we have $N \models A$ if and only if $N \models_{\preceq} A$. A monotone theory has the property (motivating the name "monotone") that every provable ground atom has a direct proof in which any term that occurs in some atom in the proof is smaller or equal to some term in its parent atom.

We will show that monotonicity is equivalent to saturation up to redundancy. Although restricting the queries may seem to reduce the usefulness of monotonicity, we show that there is no significant difference as a Horn query may be transformed to an equivalent atomic query over an extended signature. We justify this with the following construction.

Let N be a Horn theory over a signature Σ and $\succ$ a partial, well-founded ordering on ground terms over Σ. We construct a presentation N' over an extended signature Σ' that will be "equivalent" to N: N' will be monotone with respect to a "closely related" ordering $\succ'$ if and only if N is local with respect to $\succ$. Since locality says something about entailment of Horn clauses $\Gamma \rightarrow A$ from N whereas monotonicity just refers to atomic consequences A, we pass Γ as an additional parameter to predicates in N'. The function symbols in Σ' are, therefore, the function symbols of Σ, the predicate symbols of Σ, together with an additional binary (right-associative) function symbol ",". For each n-ary predicate p, Σ' contains an $n+1$-ary predicate p'. The extra argument in p' serves to pass the antecedent Γ. In addition, Σ' contains a binary predicate member to denote membership in lists constructed from ",".

Let A [T] denote the set of ground atoms [terms] over Σ. In Σ' the atoms in A are terms. To distinguish Σ-atoms from Σ-terms when considered as terms over Σ', in Σ' we assume a three-sorted type discipline where A and T are two of the sorts, and where S is an additional sort, denoting A-sequences. The predicates p' have typing $p' \subseteq S \times T \times \ldots \times T$, and $\mathsf{member} \subseteq A \times S$. "," is of type $A \times S \rightarrow S$ and the predicates p and functions f in Σ, respectively, as functions symbols in Σ' have the type $p : T \times \ldots \times T \rightarrow A$ and $f : T \times \ldots \times T \rightarrow T$.

The clauses of N' are the following:

$$
\begin{array}{rcl}
p'_1(s,\vec{t}_1),\ldots,p'_n(s,\vec{t}_n) & \rightarrow & p'_0(s,\vec{t}_0), \\
 & & \text{for } p_1(\vec{t}_1),\ldots,p_n(\vec{t}_n) \rightarrow p_0(\vec{t}_0) \text{ in } N \\
\mathsf{member}(p(x_1,\ldots,x_n),s) & \rightarrow & p'(s,x_1,\ldots,x_n), \\
 & & \text{for any } n\text{-ary predicate symbol } p \text{ in } N \\
 & \rightarrow & \mathsf{member}(a,(a,s)) \\
\mathsf{member}(a,s) & \rightarrow & \mathsf{member}(a,(b,s))
\end{array}
$$

On Σ' we consider the smallest ordering $\succ'$ containing $\succ$ and satisfying $(a,t) \succ' a$, $(a,t) \succ' t$, and $p(t_1,\ldots,t_k) \succ' t_j$, for all $p : T \times \ldots \times T \rightarrow A$ in Σ'.

The following properties are immediate by construction:

Lemma 4 *(i) Let $C = p_1(\vec{t}_1),\ldots,p_n(\vec{t}_n) \rightarrow p_0(\vec{t}_0)$ be a ground clause over Σ. Then $N \models C$ if and only if $N' \models p'_0((p_1(\vec{t}_1),\ldots,p_n(\vec{t}_n)),\vec{t}_0)$.*

(ii) N' is monotone with respect to $\succ'$ if and only if N is local with respect to $\succ$

(iii) If Σ-terms of size n have at most $f(n)$ smaller terms under $\succ$, then Σ'-terms of size n have at most $O(max(n, n * f(n)))$ smaller terms under $\succ'$.

The significance of the lemma lies in the fact that from now on it will suffice to characterize monotonicity in terms of saturation by ordered resolution. For Horn clauses, monotonicity and locality are essentially equivalent concepts.

5.2. Monotonicity implies Saturation

The definition of redundancy that we have given in section 3. was based on the notion of entailment. Entailment then referred to *all* models of a set of clauses. In the context of Horn clauses a stronger form of entailment, based on the minimal model semantics [11], is equally important and useful. Throughout this section, entailment $\models$ will now only refer to the *minimal model*. That is, a Horn clause C is entailed by a set of Horn clauses N either if N is inconsistent or else if C is true in the minimal model of N. Under this new definition of entailment, a ground clause C is redundant in N if and only if either N_C is inconsistent, or else C is true in the minimal model of N_C. By basing redundancy on a particular model, rather than on all models, more inferences and clauses become redundant. This will be the key for showing that monotonicity and saturation are equivalent.

Theorem 4 *Let N be a set of clauses and let $\succ$ be a well-founded partial ordering on terms. Then N is monotone with respect to $\succ$ if and only if N is saturated up to redundancy with respect to $\succ$.*

Proof. The "if" direction of the theorem is proved similar to the proof of Theorem 2. Let us now consider the reverse direction and assume that N is monotone with respect to

$\succ$. Moreover, we may assume that N does not contain the empty clause since otherwise N would be trivially saturated. Let $\succ'$ be any well-founded, total extension of $\succ$ on ground terms. We have to choose an atom ordering compatible with $\succ'$ (again denoted by $\succ'$) for which we will be able to prove that N is saturated with respect to $\succ'$. The definition of $\succ'$ proceeds in parallel to the inductive definition of a Herbrand interpretation J, which we later shall see is the minimal model of N.

Formally we apply induction over $\succ'$ (on terms) to define Herbrand interpretations J_s and J^s for all ground terms s as follows: J_s is the set $\bigcup_{s \succ' t} J^t$. For the construction of J^s let N^s be the set of ground instances of N in which s occurs and is the maximal term with respect to $\succ'$. By P^s we denote the subset of N^s in which s occurs positively. Moreover let T denote the standard "T_P"-operator used to define the semantics of Horn programs [11]. Then we define

$$J^s = \bigcup_{n \geq 0} T^n_{P^s}(J_s)\,.$$

In parallel we define $\succ'$ on ground atoms inductively such that it is compatible with the term ordering $\succ'$ and reflects the logical dependencies between true atoms as given by the T_P-operator. Formally, let A and B be in J^s with s the maximal term in both A and B. Then there exist (minimal) indices $n, m \geq 1$ such that A is in $T^n_{P^s}(J_s) \setminus T^{n-1}_{P^s}(J_s)$ and B is in $T^m_{P^s}(J_s) \setminus T^{m-1}_{P^s}(J_s)$. If $n > m$ then $A \succ' B$ and when $n = m$ then we order A and B in an arbitrary way. When the ground atoms A and B do not share a maximal term, then we set $A \succ' B$ iff the maximal term in A is greater than the maximal term in B with respect to $\succ'$.

Now that we have extended $\succ'$ to atoms, we can construct a second, essentially equivalent family of partial interpretations I_C, for all ground instances C of N, by induction over the extension of $\succ'$ from atoms to clauses. Formally we define, for each such C, sets of ground atoms I_C and E_C such that I_C is $\bigcup_{C \succ D} E_D$. Furthermore, if C is a clause $\Gamma \rightarrow A$, where A is strictly maximal with respect to Γ, such that C is false in I_C then $E_C = \{A\}$. In that case, we also say that C is *productive* and that it *produces* A. In all other cases, $E_C = \emptyset$. Finally, let I be $\bigcup_C E_C$.

In what follows we shall also use the notation I^C for $I_C \cup E_C$. Hence I_C is the partial interpretation defined by clauses smaller than C, whereas I^C additionally includes the effect of C in this construction. Moreover, for any ground term s, let I^s [I_s] denote the union of the E_C for all clauses C with a maximal term t such that $s \succeq' t$ [$s \succ' t$].

For all ground terms s and ground instances C of N with maximal term s we shall now prove (under the assumption that N is local) the following facts:

(i) $I_s = J_s$ and $I^s = J^s$.

(ii) I_C is the minimal model of N_C.

(iii) if C is a clause in N^s of the form $\Gamma \rightarrow A$ such that $A \succ' \Gamma$ then C is true in I^C.

(iv) if C is a clause in N^s of the form $A, \Gamma \rightarrow \Delta$ such that A is maximal with respect to Γ, Δ (in $\succ'$), then C is true in I_C.

We prove these facts simultaneously by induction over $\succ'$ on terms s with an inner induction over $\succ'$ on clauses with maximal term s. Hence we may assume that (i)–(iv) are true for terms t such that $s \succ' t$.

Ad (i): $I_s = J_s$ follows from the induction hypothesis for (i) according to which $I^t = J^t$ for all terms t such that $s \succ' t$. In the construction of I^s only clauses C in which the head A is strictly maximal may contribute to I. Then C is in P^s. C is productive if, in addition, any of its body atoms is true already in I_C. Similarly, by the construction of $\succ'$ a clause C contributes to J^s via the T_P-operator if only if its its head A is greater than any of its body atoms B with respect to $\succ'$, and if any of the body atoms have been produced in previous iteration steps (from clauses smaller than C in $\succ'$). Therefore, a clause is productive for I^s if and only if it is productive for J^s.

Ad (ii): We may use the induction hypothesis for (iii) and (iv) to infer that any ground instance D of N with $D \prec' C$ is true in I_C. Hence I_C is a model of N_C. Its minimality follows from the fact that atoms are produced into N_C only if there exists a clause producing it.

Ad (iii): Suppose C is false in I^C. Then Γ is true, but A is false in I^C. But then C would have to be productive, which is a contradiction.

Ad (iv): First, suppose that s occurs in Δ. Then C, as a clause of P^s, is true in J^s. Suppose C is false in I_C. Then A, Γ is true in I_C, hence Δ is an atom B which is true in J^s. Since $J^s = I^s$, B must be contained in I^s, and hence in I_C which is a contradiction. (In fact, the extension of $\succ'$ to clauses is such that clauses $D \succeq' C$ cannot produce atoms $B \preceq' A$ into I.)

Otherwise, assume that s occurs negatively, but not positively in C, and that C is false in I_C, hence A, Γ is true in I_C. By induction hypothesis for (iii) I_C is the minimal model of N_C. Therefore $N \models I_C$, and hence $N \models \Delta$. If Δ is empty and N is local, N contains the empty clause, which is a contradiction. Otherwise, $\Delta = B$, and if t is the maximal term in B, then $s \succ' t$, hence $t \not\succeq s$. As N is local, B follows from ground instances of N in which the maximal terms (with respect to $\succ$) are smaller or equal (with respect to $\succ$) than t. These clauses are all in N_C. Hence B is true in I_C, which is a contradiction. We may infer that C is true in I_C.

Having shown (i)–(iv), it is easy to see that N is saturated with respect to $\succ'$. By (iv), all ground instances of N that may possibly occur as the second premise in an ordered resolution inference are redundant, hence so is the inference. □

The preceding result is with respect to the notion of redundancy based on the minimal model semantics of Horn clauses. It is easy to see that if a set of Horn clauses is saturated up to redundancy with respect to the loose (all models) semantics then it is in particular also saturated with respect to the minimal model semantics.

6. Complexity Bounds for Entailment Problems

Order locality provides a new method for establishing complexity bounds for entailment of ground clauses. For the remainder of the paper we assume that clause sets under consideration are finite. If a clause set is saturated under an ordering where ground atoms have only finitely many smaller atoms that may be enumerated in finite time, then decidability of the ground entailment problem is obtained by enumerating an appropriate finite set of ground instances of the clauses. Let the size of a clause be the sum of the sizes (the number of nodes in its tree representation) of the terms in the clause. In the Horn case we have:

Theorem 5 *Let N be a set of Horn clauses that is saturated up to redundancy with respect to $\succ$. Then for ground Horn clauses C, the entailment problem for $N \models C$ is decidable in time $O(f(m)^k + g(m))$ where m is the size of C and each term of size n has $O(f(n))$ smaller terms (under $\succ$) which may be enumerated in time $g(n)$. Furthermore, the constant k depends only on N.*

Proof. As N is saturated with respect to any total extensions of $\succ$, then by Theorem 2, N is local with respect to $\succ$. Hence $N \models C$ iff $N \models_{\preceq} C$. The set of ground instances N^C of N in which terms are smaller than or equal to some term in C may be generated first by enumerating all terms bounded by terms in C and after using these to instantiate clauses in N to form the ground clauses in N^C. The enumeration step requires $O(g(m))$ time and instantiation step requires time at most $O(f(m)^k)$ where k is a constant that depends on N only; in particular, an upper bound for k is the maximum number of variables in any clause in N. A tighter bound can be found using analysis similar to McAllester's "refined tractability theorem" in [12]. By lemma 1, we can decide $N^C \models C$ in time linear in the sizes of N^C and C. □

For full clauses we obtain a similar result by reduction to the satisfiability problem for $O(f(m)^k)$ ground clauses of bounded size:

Theorem 6 *Let N be a set of full clauses that is saturated up to redundancy with respect to $\succ$. The disentailment problem $N \not\models C$ is non-deterministically decidable in time $O(f(m)^k + g(m))$, where, as before, m is size of C, f and g bound the number of smaller terms and their enumeration time, and k is a constant depending on N.*

As a special case, we obtain polynomial bounds in the Horn case if orderings are *polynomially bounded*, that is, each ground term has only polynomially many smaller ones which can be enumerated in polynomial time. McAllester's tractability result (his 'Refined Tractability Lemma' in [12]) is a special case of ours for Horn clauses and the subterm ordering, where each term has linearly many subterms which can be directly enumerated. For full clauses, locality with respect to a polynomially bounded ordering means that ground entailment is in co-NP.

We obtain exponential (respectively co-nondeterministic exponential) time bounds for entailment for Horn (respectively full) clauses if orderings are employed for which terms have at most exponentially many smaller ones, and which can be enumerated in exponential time. An example of such an *exponentially bounded ordering* is the Knuth-Bendix ordering (KBO) [10]. It has been shown in [8] that under certain restrictions decreasing chains of ground terms can be of at most exponential length if $\succ$ is a KBO. Since KBOs are total orderings (for total precedences) this implies that there are at most exponentially many smaller terms for any given ground term. Double exponentially bounded orders can be constructed using KBOs based on polynomial interpretations (where decreasing chains of ground terms have double exponential length bounds [8]) rather than just linear, weight-defined term interpretations.

7. Characterizing Complexity Classes

In this section we will restrict ourselves again to Horn clauses. We have seen that the entailment problem for an order local clause set belongs to a complexity class characterized by the term ordering. Here we prove the converse: when a ground entailment problem is decidable in polynomial (exponential, ...) time, then there is a Horn clause set defining that entailment problem which is local under a polynomially (exponentially, ...) bounded ordering. This result is in the spirit of results in descriptive complexity theory, for example that Datalog programs precisely capture the set of queries on a finite ground database decidable in finite time [9, 15]. Our proof is based on a direct Turing machine encoding where the ordering reflects machine transitions. In this way it is conceptually simpler than similar results for polynomial time queries proven in [7], which are based on transforming a problem to one involving Datalog programs.

Let a Turing machine $M = \langle Q, \Delta, \delta, F \rangle$ be given. We construct a signature and a Horn clause set R_M which recognizes the language $L(M)$ as follows. The signature contains the constant q for each state $q \in Q$, the unary function a for each letter $a \in \Delta$ (which we apply by juxtaposition, e.g., at instead of $a(t)$), a new constant ϵ, a function symbol c of arity 4 for representing configurations, and a unary predicate symbol tm. A Turing machine configuration is represented

by the ground term $c(w_m w_{m-1} \ldots w_1 \epsilon, q, a, v_1 v_2 \ldots v_n \epsilon)$, which indicates (i) initial tape content, (ii) state, (iii) current imput symbol, and (iv) remaining tape content. For each possible transition there are corresponding Horn clauses in R_M. For example if δ stipulates that M in state q reading a writes b, moves left, and enters state q' then we would have a clause

$$tm(c(W, q', a', bV)) \rightarrow tm(c(a'W, q, a, V)),$$

which formalizes the relationship between the successor configuration (on the left) and the predecessor (on the right). There are additional clauses, $\rightarrow \quad tm(c(W, f, A, V))$, for each $f \in F$, which reflect accepting computations.

Given a string x, let $\lceil x \rceil$ denote the representation of the initial configuration with respect to x. By construction, R_M *recognizes the language* $L(M)$, i.e., for every $x \in \Delta^*$, $x \in L(M) \quad \leftrightarrow \quad R_M \models tm(\lceil x \rceil)$.

Lemma 5 *If M is uniformly terminating (all computations from any initial configuration terminate), then R_M is local with respect to the ordering $\succ_M$ defined by $c(v, q, a, w) \succ_M c(v', q', a', w')$ iff there is a non-empty sequence of transitions for M from the configuration represented by $v(q, a)w$ to $v'(q', a')w'$.*

Proof. $\succ_M$ defines a well-founded partial order. For any compatible atom ordering, the right-hand side of each clause always contains the maximal atom. Hence there are no ordered resolution steps. □

When M is deterministic, the length of the unique computation history starting from the initial configuration of a string x is precisely the number of terms smaller than x under $\succ_M$. Hence, we have:

Lemma 6 *Let M be a deterministic Turing machine whose running time for all input strings x is bounded by $f(|x|)$. Then there is an ordering $\succ$, in which each ground term t has at most $f(|t|)$ $\succ$-smaller terms and a $\succ$-local clause set R which recognizes $L(M)$.*

Note that if M were nondeterministic, then terms would have $O(k^{f(|x|)})$ $\succ$-smaller terms where k is the maximum number of δ-transitions for any pair of $q \in Q$ and $a \in \Delta$.

Theorem 7 *Let R be a Horn theory over a finite signature for which ground entailment is polynomial time decidable. Then there is a Horn theory S over an extended signature that (1) is order local under a polynomial ordering and (2) for all Horn clauses C over the signature of R, $R \models C \quad \leftrightarrow \quad S \models C$.*

Proof. If the entailment problem for R is polynomial time decidable then there is some deterministic Turing machine M that operates on strings (polynomially) encoding ground Horn clauses and accepts exactly those entailed by R. Further, the time required for any computation is bounded by a polynomial. Using Lemma 6 it follows that that there is a Horn clause theory S that is local under a polynomial ordering and recognizes $L(M)$.

To finish the proof requires extending the signature and some metaprogramming. We augment the signature of S with declarations for R; we assume that the names of representatives in S are distinct from the functions and predicates in R which they represent. The resulting theory, however, doesn't prove the same Horn clauses C as R but instead C encoded, using the tm predicate, as the appropriate initial Turing machine configuration. If S is to accept C itself we must add rules for a metaprogram such that for each C, $S \models tm(...) \rightarrow C$ where $tm(...)$ is the corresponding initial configuration for C in the theory S. This requires adding rules to S for a Horn metaprogram which translate clauses C to configurations $tm(...)$. This can be done in linear time while preserving locality under a polynomial ordering. □

For concreteness we have proven the theorem for polynomial time. But we can uniformly replace the word "polynomial" by exponential or any other large complexity class and the proof remains valid. Hence polynomial (exponential, etc.) time entailment corresponds precisely to locality under the corresponding orderings.

8. Experience with Saturate

Theorem provers based on ordered resolution, such as our Saturate system [6], may be employed as (necessarily incomplete) procedures for detecting if a clause set is saturated and for saturating clause sets. What is required is that the prover supports appropriate classes of orderings and that it provides strong enough techniques for detection of redundancy (ours are based on [14]), which are necessarily approximative in nature since redundancy is undecidable. Among the two alternatives for redundancy that we have discussed in the paper, Saturate implements the one that is based on entailment with respect to all models of a set of clauses. In the context of refutational theorem proving, this is the preferred choice as it is suitable for the non-Horn case. Moreover, entailment with respect to all models is monotone, that is, is preserved under the addition of clauses to a given theory (as well as under removal of redundant clauses). This is essential in the theorem proving context.

We were able to take the Saturate system and, with very few additions, saturate a number of non-trivial examples, including congruence closure, tree embeddings, various fragments of propositional logic, and total orderings, cf. [4].

As explained in Section 4., in practice we work with Theorem 2 rather than Lemma 3. That is, we simultaneously

saturate with respect to all total extensions of a given term ordering $\succ$, employing a certain lexicographic extension of these term orderings to atoms. This requires solving ordering constraints with respect to families of orderings. One may use for this purpose a constraint solver like a specialization of the one for lexicographic path orderings previously available in Saturate, which is based on the algorithm in [13].

8.1. Congruence Closure

Consider the congruence closure example from the introduction. One presentation of this theory, over a binary function symbol f, is the following Horn clause set.

$$\begin{aligned} &\rightarrow && x = x && (1)\\ x = y &\rightarrow && y = x && (2)\\ x = y,\ y = z &\rightarrow && x = z && (3)\\ x = y,\ u = v &\rightarrow && f(x,u) = f(y,v) && (4) \end{aligned}$$

This presentation is subterm local, and McAllester's system is able to verify it. When given to Saturate, saturation terminates with these above clauses and the additional clause

$$x = y,\ u = v,\ f(x,u) = z \rightarrow f(y,v) = z \quad (5)$$

as final saturated set. Saturate generates the clause 9 due to the weaker of the two forms of redundancy that it implements. This behaviour was observed in several examples of subterm local Horn theories. While Saturate was able to saturate many of them in finite time, the final system usually contained additional clauses which are either not redundant in the weaker sense of redundancy, or the redundancy of which Saturate was not able to show. In this regard, the present version of Saturate is weaker than McAllester's locality checking procedures.

The following variation of the congruence closure example indicates one of the benefits of the saturation-based approach to locality which is that saturation may transform a given theory presentation from one that is not local into one which is. Consider

$$\begin{aligned} &\rightarrow && x = x && (6)\\ x = y &\rightarrow && y = x && (7)\\ x = y,\ y = z &\rightarrow && x = z && (8)\\ x = y &\rightarrow && f(x,z) = f(y,z) && (9)\\ x = y &\rightarrow && f(z,x) = f(z,y)\ . && (10) \end{aligned}$$

This presentation is not subterm local. For instance,

$$a = b,\ c = d \rightarrow f(a,c) = f(b,d)$$

cannot be derived without employing an intermediate term $f(a,d)$ or $f(b,c)$, neither of which is a subterm of the query

Saturate identifies that this is not local under the subterm ordering and terminates after adding the following new clauses.

$$\begin{aligned} x = y,\ f(y,z) = u &\rightarrow && f(x,z) = u && (11)\\ x = y,\ f(z,y) = u &\rightarrow && f(z,x) = u && (12)\\ x = y,\ u = v &\rightarrow && f(x,u) = f(y,v) && (13)\\ x = y,\ u = v,\ f(y,v) = z &\rightarrow && f(x,u) = z && (14) \end{aligned}$$

Note that if Saturate did a more sophisticated backward checking of redundancy, then clauses 13–16 could be deleted after the last two clauses are generated. Using Theorem 5 we can conclude that congruence closure is decidable in time $O(n^3)$, which is a good bound, although not optimal.

8.2. Propositional Calculus

We first formalize a tractable fragment of propositional logic and then the full calculus. Our tractable fragment, formalized as a set of Horn clauses, is a slight modification of [12].

$$\begin{aligned} true(p) &\rightarrow && true(or(p,q))\\ true(p) &\rightarrow && true(or(q,p))\\ true(or(p,q)), true(not(p)) &\rightarrow && true(q)\\ true(or(p,q)), true(not(q)) &\rightarrow && true(p)\\ true(not(p)), true(not(q)) &\rightarrow && true(not(or(p,q)))\\ true(not(or(p,q))) &\rightarrow && true(not(p))\\ true(not(or(p,q))) &\rightarrow && true(not(q))\\ true(not(p)), true(p) &\rightarrow && true(\mathsf{ff})\\ true(\mathsf{ff}) &\rightarrow && true(p)\\ true(not(not(p))) &\leftrightarrow && true(p) \end{aligned}$$

This is a theory presentation that is not subterm local. In fact, to derive $true(or(not(p),q)),\ true(p) \rightarrow true(q)$ requires $true(not(not(p)))$ as an intermediate atom. It is, however, saturated with a refinement of the subterm ordering[2] over the precedence $or \succ not \succ \mathsf{ff}$ and hence ground entailment is decidable in polynomial time. In this ordering $or(not(p),q)$ is greater than $not(not(p))$.

Now consider the following presentation of full propositional logic.

$$\begin{aligned} true(A) &\rightarrow && true(or(A,B))\\ true(A) &\rightarrow && true(or(B,A))\\ false(A), false(B) &\rightarrow && false(or(A,B)) \end{aligned}$$

[2] Our refinement of the subterm ordering $\rhd$, based on a precedence π on function symbols, is the following: $s \rhd_\pi t$ iff (i) $s \rhd t$, or else (ii) $s = f(\ldots)$, $t = g(t_1,\ldots,t_n)$, $s \rhd t_i$, for $1 \le i \le n$, and $f >_\pi g$. In this ordering each term can have at most $O(|t|^n)$ smaller terms, where n is the maximal arity of function symbols.

$$
\begin{aligned}
\rightarrow\ & true(A), false(A) \\
false(not(A)) \rightarrow\ & true(A) \\
true(A) \rightarrow\ & false(not(A))
\end{aligned}
$$

The presentation is trivially saturated under the subterm ordering, hence we obtain CoNP as an upper bound for the entailment (tautology) problem for propositional logic.

9. Comparison and Conclusion

Our work generalizes subterm locality both to incorporate term-orderings and to handle full clauses. Even for polynomial time decidable theories, the flexibility offered by admitting orderings other than the subterm ordering is substantial. In [4] we give examples of polynomially bounded orderings related to, but different from the subterm ordering, together with natural examples of theories that are local with respect to this ordering but which cannot be saturated finitely with respect to the subterm ordering. Our ordering-based approach also allows us to characterize entailment problems with respect to two complexity hierarchies: deterministic and co-non-deterministic time.

As an operational tool for testing order locality, we have proposed ordered resolution up to redundancy, and we show that in many cases it is equivalent to order locality. We lack space to compare ordered resolution with the procedures proposed by McAllester [12, 7] for detecting subterm locality. Roughly speaking, his are based on specialized forward and backchaining procedures on clauses augmented with extra "book-keeping information", which supports termination analysis. This method is related to the notion of redundancy for Horn clauses that is based on minimal model semantics. In our work, we have made a direct connection with standard techniques from resolution theorem proving, which enables us to use already implemented proof-procedures. Aside from providing a practical means for computer-assisted complexity analysis, this allows us to bring standard improvements, e.g., AC unification and the like, into our framework. Such generalizations, and implementing a redundancy checker based on minimal-model semantics exploiting the constructions in the proof of Theorem 4, are directions for future work.

Lemma 1 is the key to implementing a fast decision procedure for Horn entailment problems: appropriate ground instances are generated and then a linear time inference algorithm is applied to the resulting ground clause set. Another direction for future work is to avoid generating ground instances of some clauses and instead build the clauses into the inference algorithm. For example, the present approach always gives a lower bound of $O(n^3)$ for any clause set which includes transitivity; hence we get a suboptimal bound for problems like congruence closure. Transitivity and more general composition laws for binary relations can be built into ground inference systems. Ordered chaining calculi have been proposed and proved complete in the presence of a similar notion of redundancy in [3]. We believe that the results of the present paper extend to these calculi.

References

[1] L. Bachmair and H. Ganzinger. On restrictions of ordered paramodulation with simplification. In M. Stickel, editor, *Proc. 10th Int. Conf. on Automated Deduction, Kaiserslautern*, volume 449 of *Lecture Notes in Computer Science*, pages 427–441, Berlin, 1990. Springer-Verlag.

[2] L. Bachmair and H. Ganzinger. Rewrite-based equational theorem proving with selection and simplification. *Journal of Logic and Computation*, 4(3):217–247, 1994. Revised version of Technical Report MPI-I-91-208, 1991.

[3] L. Bachmair and H. Ganzinger. Rewrite techniques for transitive relations. In *Proc. 9th IEEE Symposium on Logic in Computer Science*, pages 384–393. IEEE Computer Society Press, 1994.

[4] D. Basin and H. Ganzinger. Automated complexity analysis based on ordered resolution. Technical Report MPI-I-95-2-006, Max-Planck-Institut für Informatik, Saarbrücken, 1995.

[5] W. Dowling and J. Gallier. Linear-time algorithms for testing the satisfiability of propositional horn formulae. *Journal of Logic Programming*, 3:267–284, 1984.

[6] H. Ganzinger and R. Nieuwenhuis. The Saturate system. Maintained on the world-wide web under URL http://www.mpi-sb.mpg.de/SATURATE/, 1994.

[7] R. Givan and D. McAllester. New results on local inference relations. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Third International Conference (KR-92)*, pages 403–412. Morgan Kaufmann Press, October 1992.

[8] D. Hofbauer and C. Lautemann. Termination proofs and the length of derivations. In N. Dershowitz, editor, *Proceedings of the 3rd International Conference on Rewriting Techniques and Applications*, volume 355 of *Lecture Notes in Computer Science*, pages 167–177. Springer-Verlag, 1989.

[9] N. Immerman. Relational queries computable in polynomial time. *Information and Control*, 68:86–104, 1986.

[10] D. E. Knuth and P. B. Bendix. Simple word problems in universal algebras. In J. Leech, editor, *Computational Problems in Abstract Algebra*, pages 263–297. Pergamon Press, 1970.

[11] R. A. Kowalski and M. H. van Emden. The semantics of predicate logic as a programming language. *Journal of the Association for Computing Machinery*, 23:733–742, October 1976.

[12] D. A. McAllester. Automatic recognition of tractability in inference relations. *Journal of the Association for Computing Machinery*, 40(2):284–303, April 1993.

[13] R. Nieuwenhuis. Simple LPO constraint solving methods. *Information Processing Letters*, 47(2):65–69, August 1993.

[14] P. Nivela and R. Nieuwenhuis. Saturation of first-order (constrained) clauses with the *Saturate* system. In C. Kirchner, editor, *Rewriting Techniques and Applications, 5th International Conference, RTA-93*, volume 690 of *Lecture Notes in Computer Science*, pages 436–440, Montreal, Canada, June 16–18, 1993. Springer-Verlag.

[15] M. Vardi. Complexity of relational query languages. In *14th Symposium on Theory of Computation*, pages 137–146, 1982.