JOURNAL OF COMPUTER AND SYSTEM SCIENCES 21, 317–353 (1980)

Complexity Results for Classes of Quantificational Formulas*

HARRY R. LEWIS†

Harvard University, Aiken Computation Laboratory, Cambridge, Massachusetts 02138

Received March 23, 1979; revised June 25, 1980

We analyze the computational complexity of determining whether F is satisfiable when F is a formula of the classical predicate calculus obeying certain syntactic restrictions. For example, for the monadic predicate calculus and the Gödel or $\exists \cdots \exists\forall\exists \cdots \exists$ prefix class we obtain lower and upper nondeterministic time bounds of the form $c^{n/\log n}$. The lower bounds are established by using acceptance problems for time-bounded Turing machines and alternating pushdown and stack automata.

1. INTRODUCTION

Most work on the complexity of logical decision problems has been focussed either on the propositional calculus [7, 21] or on decidable theories such as Presburger arithmetic [4, 11, 25] or various theories of successor and ordering [24, 31]. Here we consider instead some subclasses of the classical predicate calculus that are defined by purely syntactic restrictions on the formation of formulas. These are formulas without function signs or the identity sign, and they are all closed, that is, contain no free variable. The four main classes we consider are the monadic predicate calculus, of which the decidability dates back to 1915 [23], and three classes determined by the form of the prefix, which are named in honor of individuals who proved their decidability:

Ackermann class: formulas with prefixes of the form [2]

$$\exists \cdots \exists\forall\exists \cdots \exists;$$

Gödel class: formulas with prefixes of the form [12]

$$\exists \cdots \exists\forall\exists \cdots \forall;$$

Schönfinkel–Bernays class: formulas with prefixes of the form [5]

$$\exists \cdots \exists\forall \cdots \forall.$$

(Any prefix other than these yields an unsolvable satisfiability problem.)

* A preliminary account of some of these results appeared in [19].

† Research supported by NSF Grant MCS76–09375–A01.

0022-0000/80/060317–37$02.00/0

We also consider a class $\mathscr{S}$ of formulas with prefixes of the form $\exists \cdots \exists\forall\exists \cdots \exists\forall$; the precise definition of this class is in Section 2A.

Our results are summarized as follows:

Monadic predicate calculus: Lower and upper bounds of the form $\mathrm{NTIME}(c^{n/\log n})$;

Gödel or $\exists \cdots \exists\forall\forall\exists \cdots \exists$ prefix class: Lower and upper bounds of the form $\mathrm{NTIME}(c^{n/\log n})$;

Schönfinkel–Bernays or $\exists \cdots \exists\forall \cdots \forall$ prefix class: Lower and upper bounds of the form $\mathrm{NTIME}(c^n)$;

Ackermann or $\exists \cdots \exists\forall\exists \cdots \exists$ prefix class: Lower and upper bounds of the form $\mathrm{DTIME}(c^{n/\log n})$ for formulas with monadic predicate letters only;

The class $\mathscr{S}$: Lower and upper bounds of the form $\mathrm{DTIME}(c^{c^{n/\log n}})$.

The only previous work in this area was by Meyer and Rackoff [28] on the monadic class. They obtained the $\mathrm{NTIME}(c^{n/\log n})$ lower bound for this class and an $\mathrm{NTIME}(c^{n^2/\log^2 n})$ upper bound. For this class our results tighten the upper bound and show that the lower bound can be obtained for the subclass of monadic formulas with Gödel prefix. Since the appearance of the preliminary version of this paper [19], Plaisted [26] has obtained some related results on formulas in conjunctive normal form (see also the Acknowledgments at the end of this paper), and Goldfarb [13] has established a non-primitive-recursive lower bound for the Gödel class with identity (which, however, is not known to be decidable).

The lower bounds for the monadic, Gödel, and Schönfinkel–Bernays classes are obtained by direct reductions of the acceptance problem for nondeterministic exponential-time bounded Turing machines. The crux of these reductions is economical representation in first-order formulas of the successor relation between encodings of natural numbers. Such representations are reminiscent of those used by Jones and Selman [15].

The lower bound for the Ackermann class is obtained by reducing to the satisfiability problem for that class the problem of whether an alternating pushdown automaton accepts an input string [6, 17]. For the class $\mathscr{S}$ we present a similar reduction for alternating stack automata [18]. A brief explanation of these automata appears in Section 2C.

The upper bounds for the Gödel and Schönfinkel–Bernays classes are obtained by means of their "finite model" property: If a formula of one of these types is satisfiable then it has a model of a predictable size. For the monadic class we design an efficient nondeterministic version of the "mini-scope" or "antiprenex" procedure. For the Ackerman class we adapt a criterion due to Kalmár [16] and Schutte [29]; for the class $\mathscr{S}$ we present a reduction, due essentially to Warren Goldfarb, to the Ackermann class.

2. PRELIMINARIES

A. *Logic*[1]

As already stated, we are concerned with the classical predicate calculus without the identity sign or functions signs. Function signs do, however, enter the picture when we pass from a formula to its *functional form.* Assume that the formula is closed, that is, has no free variable. If no quantifier lies within the scope of a negation sign, then the functional form is constructed thus:

1. Rename variables so that no two quantifiers quantify the same variable;

2. Choose a unique function sign for each existentially quantified variable, and replace each occurrence of that variable (except for the one immediately after the existential quantifier) by the term $f(y_1,...,y_n)$, where f is the chosen function sign and $y_1,...,y_n$ are the variables universally quantified by quantifiers in whose scopes the existential quantifier lies;

3. Delete all quantifiers.

For example, the functional form of $\forall yPy \wedge \forall y\exists xPx \wedge \exists zQz$ might be $Py_1 \wedge Pf(y_2) \wedge Qa$, where f is a monadic (one-place) function sign and a is a constant sign (zero-place function sign).[2] (If some quantifiers are within the scopes of negation signs, the same procedure is followed except that a universal quantifier in the scopes of an odd number of negation signs is treated as an existential quantifier, and an existential quantifier in the scopes of an odd number of negation signs is treated as a universal quantifier.)

The importance of the functional form arises from its use in the definition of Herbrand expansion. Call the set of all terms constructible from the function signs in the functional form the *Herbrand universe.* (A constant sign has to be added as "seed" if there is none in the functional form.) The *Herbrand expansion* is then the set of all instances of the functional form that can be obtained by substituting terms from the Herbrand universe for the variables of the functional form. For example, $Pa \wedge Pf(a) \wedge Qa$ and $Pf(f(a)) \wedge Pf(a) \wedge Qa$ are in the Herbrand expansion of the formula $\forall yPy \wedge \forall y\exists xPx \wedge \exists zQz$ used as an example just above.

EXPANSION THEOREM (Skolem–Herbrand–Gödel). *A formula is satisfiable if and only if its expansion is satisfiable (as a set of formulas of the propositional calculus).*

We write $D(F)$ for the Herbrand universe of a formula F. The term "instance" is reserved to mean "instance obtained by substituting terms from the Herbrand universe for the variables," so that the Herbrand expansion of a formula is the set of all instance of its functional form.

We adopt the following notation for substitution: $F[y_1/t_1,...,y_n/t_n]$ is the result of

[1] Further discussion of Herbrand expansions may be found in [10, 22].

[2] In general, the symbols x, y, z, with or without subscripts or primes, are variables; a, b, c, d, with or without subscripts, are constant signs.

substituting term t_1 for all occurrences of variable y_1 in formula F,..., t_n for all occurrences of y_n, all substitutions being carried out simultaneously. For example, $Py_1y_2[y_1/y_2, y_2/f(y_1)] = Py_2f(y_1)$; and if G is the functional form of a formula F and G has variables $y_1,..., y_n$, then the Herbrand expansion of F is $\{G[y_1/t_1,..., y_n/t_n]: t_1,..., t_n \in D(F)\}$.

We use $\leftrightarrow$ as a symbol for the biconditional in our logical object language, and $\equiv$ as a symbol for logical equivalence in the metalanguage. The sign $=$ is part of the metalanguage only.

We also adopt the following abbreviations. *Monadic* is the class of formulas with only monadic (one-place) predicate letters. Regular expressions are used to denote classes of formulas determined by their prefixes; for example $\exists^*\forall\exists^*$ is the set of all prenex formulas of the form $\exists z_1 \cdots \exists z_k \forall y \exists x_1 \cdots \exists x_m F$, where k, $m \geqslant 0$, $z_1,..., z_k, y, x_1,..., x_m$ are variables, and F is quantifier-free. If S is any set of formulas, S-Sat is the set of formulas in that class that are satisfiable. For example, $\exists^*\forall\forall\exists^* \cap$ Monadic-Sat is the set of satisfiable formulas of the Gödel class with monadic predicate letters only.

The class $\mathscr{F}$ is an extension of the solvable class J of [9] (see also [10, 18]). It is the class of all formulas with prefixes of the form $\exists z_1 \cdots \exists z_k \forall y_1 \exists x_1 \cdots \exists x_m \forall y_2$, containing dyadic predicate letters only, and not having any atomic subformula of the form Py_2y_1 or Py_2x_j $(j = 1,..., m)$. (Allowing either of these forms yields an unsolvable class, even if $k = 0$ and $m = 1$ [1, 20].) Its solvability is of some interest, since it is one of the few solvable classes known that contain satisfiable formulas without finite models. An example of such a formula in $\mathscr{F}$ is

$$\forall y_1 \exists x_1 \forall y_2 (Px_1y_1 \wedge (Py_1y_2 \rightarrow Px_1y_2) \wedge \neg Py_1y_1).$$

B. *Computational Complexity*

Our model of computation is the Turing machine with a single one-way infinite tape used both for presenting the input and for subsequent computations. We use $|x|$, for any object x, to denote the length in symbols of some natural encoding of x. Thus $|x|$ is the length of x in symbols, if x is a string over a fixed alphabet; $|\$x_1\$x_2 \cdots \$x_n\$|$ if $x = \{x_1,..., x_n\}$ is a finite set of strings; and $\log x$, if x is a natural number represented in binary. (All log's are base 2, unless otherwise specified.) The following complexity classes will be used:

$$\text{DTIME}(f(n)) = \{S: S \text{ is accepted by a deterministic Turing machine operating in time bound } f(n)\};$$

$$\text{NTIME}(f(n)) = \{S: S \text{ is accepted by a nondeterministic Turing machine operating in time bound } f(n)\};$$

$$\text{DEXPTIME} = \bigcup_{c>0} \text{DTIME}(c^n);$$

$$\text{NEXPTIME} = \bigcup_{c>0} \text{NTIME}(c^n);$$

$$\text{D2EXPTIME} = \bigcup_{c>0} \text{DTIME}(c^{c^n}).$$

Let $S_1 \subseteq \Sigma^*$ and $S_2 \subseteq \Gamma^*$, where Σ and Γ are finite alphabets. S_1 is said to be "efficiently" reducible to S_2 if there is an "efficient" mapping $f: \Sigma^* \to \Gamma^*$ such that $x \in S_1$ if and only if $f(x) \in S_2$. By "efficient" we mean here "computable in deterministic polynomial time"; in most cases these reductions can also be seen to require only logarithmic work space, but we neither demonstrate nor use this sharper notion of "efficiency." If S_2 is in DEXPTIME, NEXPTIME, or D2EXPTIME then we are guaranteed that S_1 is in the same class only if the reduction causes not more than a linear increase in the length of the input. For this reason we sharpen the notion of "efficient reducibility" thus: We say that S_1 is *reducible to* S_2 *via length order* $g(n)$ if S_1 is in deterministic polynomial time reducible to S_2 by means of a mapping f such that, for some $c > 0$, $|f(x)| \leqslant cg(|x|)$ for all x. Furthermore, if $\mathscr{C}$ is one of the classes listed above, then we say that $\mathscr{C}$ *is reducible to* S *via length order* $g(n)$ if each $S' \in \mathscr{C}$ is reducible to S via length order $g(n)$. In the particular cases in which $\mathscr{C}$ is DEXPTIME, NEXPTIME, or D2EXPTIME we use such reductions in conjunction with deterministic and nondeterministic time hierarchy results [8, 14, 30] to establish lower bounds in the following way:

PROPOSITION 2.1. *Let g and h be functions such that for every $c_1 > 0$ there is a $c_2 > 0$ such that for all n, $h(c_1 g(n)) \leqslant c_2 n$. Suppose that* DEXPTIME, NEXPTIME, *or* D2EXPTIME *is reducible to S via length order $g(n)$. Then there is a $c > 1$ such that $S \notin \text{DTIME}(c^{h(n)})$, $S \notin \text{NTIME}(c^{h(n)})$, or $S \notin \text{DTIME}(c^{c^{h(n)}})$, respectively.*

Proof. We consider just the case of NEXPTIME, the other proofs being strictly analogous.

Let A be any set in NEXPTIME that is not in (say) NTIME(3^n). If S were decidable nondeterministically in time $c^{h(n)}$, then since A can be reduced to S with a length increase of at most $c_1 \cdot g(n)$ for some c_1, S could be decided nondeterministically in time $c^{h(c_1 g(n))} \leqslant c^{c_2 n}$. By hypothesis this is impossible unless $c^{c_2} > 3$, so if $c < 3^{1/c_2}$ then $S \notin \text{NEXPTIME}(c^n)$. ∎

This Proposition is used in this paper with the following g and h:

$$g(n) = n \log n; \qquad h(n) = n/\log n,$$

$$g(n) = n; \qquad h(n) = n.$$

In keeping with usual practice in the theory of computation, we regard all strings in a class—and in particular, in a class of first-order formulas—as composed of only finitely many distinct symbols. This means, for example, that if S is a class of first-order formulas, and formulas in S may contain arbitrarily many predicate letters, then there is a constant b such that a formula with k distinct predicate letters must asymptotically have length at least $k \log_b k$. Put another way, a formula of length n

may have only $O(n/\log n)$ predicate letters. For if $k \geqslant dn/\log n$ then $\log k > \log n - \log\log n > (\log n)/2$, so that $k \log k > (dn/\log n) \cdot (\log n)/2 = dn/2$. Since[3] $n = \Omega(k \log k)$, for each d there can be only finitely many n such that $k \geqslant dn/\log n$.

Another consequence of this assumption is that the disjunctive normal form of a formula of length n is asymptotically of length $c^{n/\log n}$ for some constant c, since such a formula can asymptotically have only $O(n/\log n)$ distinct atomic formulas.

C. *Alternating Automata*

In Sections 9 and 11 we shall reduce DEXPTIME to the satisfiability problem for the Ackermann class and D2EXPTIME to the satisfiability problem for the class $\mathscr{F}$. Our method is to use characterizations of these complexity classes established in [17]: A set is in DEXPTIME if and only if it is the language accepted by some two-way alternating pushdown automaton, and is in D2EXPTIME if and only if it is the language accepted by some two-way alternating stack automaton. (The fact that every set in DEXPTIME is accepted by some alternating pushdown automaton was first observed by Chandra and Stockeyer [6].) What follows is an intuitive account of these automata and a formalization of their structure. (Our formalizations are rather different from those of [17], though they are readily seen to be equivalent. Our goal in using a different formalization is to simplify the subsequent constructions.)

Pushdown Automata. A two-way pushdown automaton has a read-only tape on which its input is presented and a pushdown store of potentially unbounded size. At the beginning of a computation the input string is placed on the input tape surrounded by the left and right endmarkers $\vdash$ and $\dashv$, which cannot appear in the string itself, and the reading head is placed over the left endmarker. The pushdown store is initialized to have a special bottom-marker Z on it. Subsequently, the automaton can move its reading head left or right; it ceases to operate if it attempts to move its reading head left off the left endmarker or right off the right endmarker. Also, it may add symbols to the top of the pushdown store ("push" them) or remove symbols from the top of the pushdown store ("pop" them) but it is not allowed to push another bottom-marker onto the pushdown store and it ceases to operate if it attempts to pop the bottom-marker off the pushdown store. The complete status of the machine at any point in time is captured in a *configuration*, which consists of the state of the finite control, the contents of the input tape and the position of the reading head on that tape, and the contents of the pushdown store. One state is designated as *final*; the input string is *accepted* if a computation leads the machine from the initial configuration to one in which the state is final.

The unique feature of an *alternating* automaton is its control structure. Certain states are designated as *universal* or *existential branching* states. From an existential branching state, the automaton is allowed to enter any one of some fixed set of other states; an accepting computation is deemed to occur if *some* choice of next state leads eventually to the final state. (These are essentially nondeterministic branches.) From a *universal* branching state, on the other hand, the automaton is required to enter

[3] The notation $f = \Omega(g)$ means that there exist c and n_0 such that $g(n) \geqslant cf(n)$ for all $n > n_0$.

each of some fixed set of states and to pursue all subsequent computations in parallel; an accepting computation is one in which *every* choice of next state leads eventually to the final state. The "alternating" character of these computations arises from the possibility that a machine may pass through a series of branching states, some of which are existential and some universal.

To be somewhat more specific, the notion of "accepting configuration" is defined recursively in the following manner. Every configuration in which the state is final is accepting. A configuration with a state that is of the existential branching type is accepting if some one of the next configurations the machine may enter (in which only the state has been changed) is accepting. And a configuration whose state is of the universal branching type is accepting if *each* of the next configurations the machine may enter is accepting. The input string is then accepted if the initial configuration is accepting.

We are now ready to be completely formal. An *alternating pushdown automaton* M consists of the following parts:

A finite set K of states,

A finite input alphabet Σ,

A finite pushdown store alphabet Γ,

Left and right endmarkers $\vdash$ and $\dashv$, not in Σ,

A designated bottom-marker Z, in Γ,

A designated start state s, in K,

A designated final state f, in K.

The set of states is divided into seven disjoint subsets. With each of these a function is associated; the (disjoint) union of these functions is the transition function of the machine. The subsets of K and the corresponding functions are as follows:

Subset of K	Name	Transition function
K_1	Read states	$\delta_1: K_1 \times (\Sigma \cup \{\vdash, \dashv\}) \to K$
K_2	Move states	$\delta_2: K_2 \to K \times \{-1, +1\}$
K_3	Push states	$\delta_3: K_3 \to K_3 \times (\Gamma - \{Z\})$
K_4	Pop states	$\delta_4: K_4 \to K$
K_5	Pushdown store read states	$\delta_5: K_5 \times \Gamma \to K$
K_6	Existential branch states	$\delta_6: K_6 \to 2^K - \{\varnothing\}$
K_7	Universal branch states	$\delta_7: K_7 \to 2^K - \{\varnothing\}$

A *configuration* is a quadruple (q, w, i, γ), where $q \in K$, $w \in \vdash \Sigma^* \dashv$, $1 \leqslant i \leqslant |w|$, and $\gamma \in \Gamma^*$. (The top of the pushdown store corresponds to the *left* end of γ.) A configuration (q, w, i, γ) is *accepting* if q is f, the final state, or if any of the following holds:

(a) $q \in K_1$ and $(\delta_1(q, a), w, i, \gamma)$ is accepting, where a is the ith symbol of w.

(b) $q \in K_2$ and $(p, w, i + \varepsilon, \gamma)$ is accepting, where $\delta_2(q) = (p, \varepsilon)$.

(c) $q \in K_3$ and $(p, w, i, X\gamma)$ is accepting, where $\delta_3(q) = (p, X)$.

(d) $q \in K_4$ and (p, w, i, γ') is accepting, where $\gamma = X\gamma'$ for some $X \in \Gamma - \{Z\}$ and $\gamma' \in \Gamma^*$, and $\delta_4(q) = p$.

(e) $q \in K_5$ and $(\delta_5(q, X), w, i, \gamma)$ is accepting, where $\gamma = X\gamma'$ for some $X \in \Gamma$ and $\gamma' \in \Gamma^*$.

(f) $q \in K_6$ and (p, w, i, γ) is accepting for some $p \in \delta_6(q)$.

(g) $q \in K_7$ and (p, w, i, γ) is accepting for each $p \in \delta_7(q)$.

M is said to *accept* the string $u \in \Sigma^*$ if and only if $(s, \vdash u \dashv, 1, Z)$ is an accepting configuration. Let $L(M) = \{u \in \Sigma^* : M \text{ accepts } u\}$; then

PROPOSITION 2.2 [17]. *$S \in$ DEXPTIME if and only if $S = L(M)$ for some alternating pushdown automaton M.*

Stack Automata. A stack automaton is very much like a pushdown automaton, except that the "stack head" that is used for adding symbols to the pushdown store or removing them may also penetrate within the pushdown store, but solely for the purpose of examining it (not changing it). When and if it subsequently returns to the top, pushing or popping may resume. A pushdown store that can be manipulated in this way is called a *stack*. An alternating stack automaton is a stack automaton with the same sort of control and acceptance structure as an alternating pushdown automaton.

Formally, an alternating stack automaton consists of the same parts as an alternating pushdown automaton, except that the state set and transition function are divided into nine parts. The first seven parts are the same as for a pushdown automaton, except that K_5 is renamed "stack read states." The two new parts are

K_8	stack head down states	$\delta_8 : K_8 \to K$
K_9	stack head up states	$\delta_9 : K_9 \to K$

A *configuration* is a quintuple (q, w, i, γ, j), where (q, w, i, γ) is a configuration as defined earlier and $1 \leqslant j \leqslant |\gamma|$. A configuration (q, w, i, γ, j) is *accepting* if q is f, the final state, or

(a) $q \in K_1$ and $(\delta_1(q, a), w, i, \gamma, j)$ is accepting, where a is the ith symbol of w.

(b) $q \in K_2$ and $(p, w, i + \varepsilon, \gamma, j)$ is accepting, where $\delta_2(q) = (p, e)$.

(c) $q \in K_3, j = |\gamma|$, and $(p, w, i, X\gamma, j+1)$ is accepting, where $\delta_3(q) = (p, X)$.

(d) $q \in K_4, j = |\gamma|$, and $(p, w, i, \gamma', j-1)$ is accepting, where $\gamma = X\gamma'$ for some $X \in \Gamma - \{Z\}$ and $\gamma' \in \Gamma^*$, and $\delta_4(q) = p$.

(e) $q \in K_5$ and $(\delta_5(q, X), w, i, \gamma, j)$ is accepting, where X is the $(|\gamma| - j + 1)$st symbol of γ.

(f) $q \in K_6$ and (p, w, i, γ, j) is accepting for some $p \in \delta_6(q)$.

(g) $q \in K_7$ and (p, w, i, γ, j) is accepting for each $p \in \delta_7(q)$.

(h) $q \in K_8$ and $(\delta_8(q), w, i, \gamma, j-1)$ is accepting.

(i) $q \in K_9$ and $(\delta_9(q), w, i, \gamma, j+1)$ is accepting.

A string $u \in \Sigma^*$ is *accepted* if and only if $(s, \vdash u \dashv, 1, Z, 1)$ is an accepting configuration. Let $L(M) = \{u \in \Sigma^* : M \text{ accepts } u\}$. Then

PROPOSITION 2.3 [17]. *$S \in$ D2EXPTIME if and only if $S = L(M)$ for some alternating stack automaton M.*

One further automata-theoretic fact we shall need is the well-known result that deterministic time classes are closed under complementation.

PROPOSITION 2.4. *Let $S \subseteq \Sigma^*$. If S is in* DEXPTIME *then so is $\Sigma^* - S$; and if S is in* D2EXPTIME *then so is $\Sigma^* - S$.* ∎

3. UPPER BOUNDS: GENERAL REMARKS

If a class of formulas contains only unsatisfiable formulas and formulas with finite models, then satisfiability is decidable for formulas in the class, since both the unsatisfiable formulas and the formulas with finite models can be recursively enumerated. (All the classes considered here have this property, called *finite controllability*, except the class $\mathscr{F}$.) This sort of decision procedure for formulas in a class $\mathscr{C}$ can be sharpened if there is a recursive "bounding function" β such that, for any $F \in \mathscr{C}$, if F is satisfiable then F has a model of cardinality at most $\beta(F)$. In fact, bounding functions are known for each class we discuss to which this method is applicable; for example, a formula F of the monadic predicate calculus has a model only if it has a model of cardinality at most 2^n, where n is the number of predicate letters of F. (For details, see the original papers, [3] or [10].)

If a class $\mathscr{C}$ has such a bounding function β, then whether a formula $F \in \mathscr{C}$ is satisfiable can be checked nondeterministically by guessing a $p \leqslant \beta(F)$ and a model of cardinality p. How complex is this procedure?

One way to implement the procedure is to form the so-called "common expansion" of F over a universe of p elements, and then to deliver that formula to a satisfiability-checker for the propositional calculus. The common expansion of F is formed by

introducing constants $a_1,\ldots,a_p$ not appearing elsewhere in F, and replacing subformulas thus:

$$\exists xG \quad \text{by} \quad \bigvee_{i=1}^{p} G[x/a_i]$$

$$\forall xG \quad \text{by} \quad \bigwedge_{i=1}^{p} G[x/a_i].$$

The constants a_i can be coded by strings of length $O(\log p)$, so if there are q quantifiers in all the common expansion of size $O(p^q \log p \cdot |F|)$, which is $O(p^{|F|})$. Since this formula may be regarded as a sentence of the propositional calculus to be passed on to a nondeterministic polynomial-time procedure, the following result is obtained:

PROPOSITION 3.1. *Whether a formula F with q quantifiers has a model of cardinality p can be ascertained nondeterministically in time*

$$c^{q\log p+\log|F|}\ (=O(c^{|F|\log p}))$$

for some constant c. ∎

In some cases this general approach can be improved by guessing the model first, rather than waiting until after the common expansion has been formed.

PROPOSITION 3.2. *Whether a prenex formula F with u universal quantifiers has a model of cardinality p can be ascertained nondeterministically in time $f(|F| \cdot p^u)$, for some polynomial f.*

(Such a bound is, like that of Proposition 3.1, in general exponential in $|F|$; but we shall apply the proposition in Section 7 to a class of prenex formulas each of which has only two universal quantifiers.)

Proof. Suppose first that we had in hand a particular structure of cardinality p which might or might not be a model for F. If it is a model for F, then for every u-tuple of substituents for the universally quantified variables there is a "correct" set of choices for the existentially quantified variables so that the formula is true in the model for these values of the variables. (Depending on the quantifier-structure of the formula, some of the choices for the existentially quantified variables may of course have to be independent of the choices of some of the universally quantified variables.) If it were possible to guess the appropriate substituents for the existentially quantified variables, then it could be confirmed that the structure at hand is a model for F simply by running through the p^u u-tuples of choices for the universally quantified variables.

But then it may not be necessary to have the entire model in hand, since the total number of atomic formulas ever checked could not exceed p^u times the number of

atomic formulas of F. The truth-values the structure takes on other atomic formulas are irrelevant to the question of whether the structure is a model for F. The following nondeterministic procedure thus emerges:

(1) Guess a structure appropriate to F of cardinality p, by writing down at most $p^u \cdot |F|$ atomic formulas with predicate letters drawn from F and arguments drawn from $a_1,..., a_p$. Any atomic formula on this list is considered "true," the rest "false."

(2) Enumerate all possible substitutions for the universally quantified variables, guessing at the appropriate moments the correct substitutions for the existentially quantified variables, and checking for each complete set of substitutions that the tabulated structure does indeed make the formula true.

The constants are of length $O(\log p)$, so each atomic formula of step (1) is of length at most $|F| \log p$. The time to generate the table in step (1) is therefore $O(|F|^2 p^u \log p)$. Similarly, the time for step (2) is $O(|F|^2 p^u \log p)$. Therefore the total time is polynomial in $|F| \cdot p^u$, as was to be shown. ∎

4. Monadic and Gödel Classes: Lower Bound

THEOREM 4.1. NEXPTIME *is reducible to* Monadic–Gödel–Sat *via length order* $n \log n$.

Proof. We present a reduction of acceptance by nondeterministic exponential-time bounded Turing mchines to formulas in this class. Let S be a set in NTIME(c^n) for some c, and let M be a nondeterministic Turing machine accepting S in this time bound. Without loss of generality, assume that c is a power of 2, and let $d = \log c$. Also without loss of generality, we may assume that M accepts an input string w of length n simply by having a computation of $c^n - 1$ steps on input w, and rejects w if there is no computation of that many steps on input w. For if M accepts instead by entering a designated final state, then M can be modified to obtain a two-tape Turing machine M' that accepts in the desired way. The machine M' uses its second tape as a clock that counts up to $c^n - 1$; if M would have accepted by final state within this time bound, then M' enters a trivial infinite loop, and otherwise M' curtails its computation before the clock assumes its maximum value. Eliminating the second tape to obtain a one-tape machine M'' that also accepts in the desired way entails squaring the time bound, but M'' also accepts S in nondeterministic exponential time since $(c^n)^2 = (c^2)^n$.

Now fix some input string w; let $|w| = n$ and $s = c^n = 2^{dn}$. We construct a formula F of length $O(n \log n)$ that is satisfiable if and only if M accepts w. (The constant implicit in O-notation depends on M and c.) The formula F is of the form $\exists z F_1 \wedge \forall y \exists x F_2 \wedge \forall y_1 \forall y_2 F_3$, where F_1, F_2, and F_3 have no quantifiers; clearly F has a prenex equivalent in $\exists^* \forall \forall \exists^*$. Actually, we present not F but the conjuncts of its functional form, in which the constant sign a has supplanted z and the term $f(y)$ has supplanted x. Thus the Herbrand expansion of F is the set of all instances obtained

from the functional form by substituting terms from the set $\{a, f(a), f(f(a)),...\}$ for the variables y, y_1, and y_2. For each $r \in \mathbb{N}$ let $\mathbf{r}$ be the term $f(f(\cdots(a)\cdots))$ with r occurrences of f in all; for example, $\mathbf{0} = a$.

Let Σ be the tape alphabet of M and K its state set, and let $\Gamma = \Sigma \cup (K \times \Sigma)$. Then a computation of length s (one with $s-1$ steps) can be pesented as a mapping μ: $\{0,..., s-1\} \times \{0,..., s-1\} \to \Gamma$. Here for each j there must be a unique i such that $\mu(i,j) \in K \times \Sigma$; in this case i specifies the head position at time j, the first component of $\mu(i,j)$ is the state at time j, and the second component is the scanned symbol at time j. And if $\mu(i,j) \in \Sigma$ then the symbol on the ith tape square (counting from 0) at time j is $\mu(i,j)$. In particular, if μ represents a computation on input w and $w = w_0 \cdots w_{n-1}$ (each $w_i \in \Sigma$) then $\mu(0,0) = (q_0, w_0)$, where q_0 is the initial state of M; $\mu(i,0) = w_i$ for $i = 1,..., n-1$; and $\mu(i,0) = \mathbf{B}$, the blank symbol, for $i = n, n+1,..., s-1$.

The basic idea is to use the terms $\mathbf{0}, \mathbf{1},..., \mathbf{s^{12}-1}$ to stand for the s^{12} sextuples $(\pi_1,..., \pi_6)$, where each of $\pi_1,..., \pi_6$ is one of the s^2 argument-pairs $(0,0),..., (s-1, s-1)$ of the putative mapping μ. The intended correspondence is that if $0 \leqslant r \leqslant s^{12}-1$, then $\mathbf{r}$ corresponds to the sextuple $((p_1, q_1),..., (p_6, q_6))$, where the $12dn$-bit binary notation for r is obtained by concatenating the dn-bit binary notations for $q_6, p_6,..., q_1, p_1$ in that order. The formula contains monadic predicate letters B_i $(i = 0,...,12dn-1)$ and S_i $(i = 1,..., 5)$ with the following intended interpretations for an argument $\mathbf{r}$ representing $((p_1, q_1),..., (p_6, q_6))$ as just described:

$B_i\mathbf{r}$ is true if and only if the ith bit of the binary notation of r is 1. (Bit 0 is the most significant bit.)

$S_1\mathbf{r}$ is true if and only if $(p_2, q_2) = (p_1 + 1, q_1)$.

$S_2\mathbf{r}$ is true if and only if $(p_3, q_3) = (p_2 + 1, q_2)$.

$S_3\mathbf{r}$ is true if and only if $(p_4, q_4) = (p_1, q_1 + 1)$.

$S_4\mathbf{r}$ is true if and only if $(p_5, q_5) = (p_2, q_2 + 1)$.

$S_5\mathbf{r}$ is true if and only if $(p_6, q_6) = (p_3, q_3 + 1)$.

Thus $S_1\mathbf{r} \wedge \cdots \wedge S_5\mathbf{r}$ is true if and only if $\mathbf{r}$ represents a sextuple of the form $((p,q), (p+1,q), (p+2,q), (p,q+1), (p+1,q+1), (p+2,q+1))$; the ability to specify such patterns enables us to specify how the state, head position, and tape contents at one instant of time depend on those at the previous instant. We begin by presenting conjuncts that serve as axioms for these predicate letters; several auxiliary predicate letters are introduced to assist in specifying these conditions.

Let $m = 12dn$. To axiomatize B_i we use monadic predicate letters B_i^* $(i = 0,..., m)$ with the intended interpretation that

$B_i^*\mathbf{r}$ is true if and only if bits $i, i+1,..., m-1$ of the binary notation for r are 1.

The interpretations of the B_i and B_i^* are fixed by the following clauses:

$$B_m^* y$$

$$\bigwedge_{i=0}^{m-1} (B_i^* y \leftrightarrow (B_{i+1}^* y \wedge B_i y))$$

$$\bigwedge_{i=0}^{m-1} \neg B_i a$$

$$\bigwedge_{i=0}^{m-1} (B_i f(y) \leftrightarrow \neg(B_i y \leftrightarrow B_{i+1}^* y)).$$

The last clause succinctly describes how the bits of a binary notation change in passing from a number to its successor. Bit i if $r+1$ is 1 if and only if either (a) bit i of r is 0 and bits $i+1,\ldots, m-1$ of r are all 1; or (b) bit i of r is 1 and some one of the bits $i+1,\ldots, m-1$ is 0.

The axiomatizations for $S_1,\ldots, S_5$ are quite similar to each other; we present the details just for S_1. We need to assert that bits $9dn,\ldots, 10dn-1$ encode the dn-bit successor of bits $11dn,\ldots, 12dn-1$, and that bits $8dn,\ldots, 9dn-1$ are the same as bits $10dn,\ldots, 11dn-1$, respectively. The conditions are ensured by the formula

$$S_1 y \leftrightarrow \bigwedge_{i=0}^{dn-1} (B_{9dn+i} y \leftrightarrow \neg(B_{11dn+i} y \leftrightarrow B_{11dn+i+1}^* y))$$
$$\wedge \bigvee_{i=0}^{dn-1} B_{9dn+i} y$$
$$\wedge \bigwedge_{i=0}^{dn-1} (B_{8dn+i} y \leftrightarrow B_{10dn+i} y).$$

We were fortunate here in having the B_i^* already available. In order to specify the other S_j, we would need to introduce additional predicate letters to signify, for example, that bits $10dn+i$ through $11dn-1$ are 1.

The remainder of the construction has two main aspects: a general description of the operation of M, independent of the input (except for its length), and a specification of the particular input string w. We begin with the former task. For each symbol $\sigma \in \Gamma$ and for $i = 1,\ldots, 6$, we have a monadic predicate letter $P_{\sigma i}$ with the following intended interpretation (where $\mathbf{r}$ represents $((p_1, q_1),\ldots, (p_6, q_6))$ as previously described):

$P_{\sigma i}\mathbf{r}$ is true if and only if $\mu(p_i, q_i) = \sigma$.

We must first ensure that $P_{\sigma i}$ and $P_{\sigma j}$ are in agreement, even when the same pair (p, q) is encoded as different components of different sextuples. For this we include the clause

$$\bigwedge_{j,k=1}^{6} \left(\bigwedge_{i=0}^{2dn-1} (B_{m-2dnj+i} y_1 \leftrightarrow B_{m-2dnk+i} y_2) \rightarrow \bigwedge_{\sigma \in \Gamma} (P_{\sigma j} y_1 \leftrightarrow P_{\sigma k} y_2) \right)$$

(This is the only part of F involving both y_1 and y_2.) Next we must ensure that μ is a function:

$$\bigoplus_{\sigma \in \Gamma} P_{\sigma 1} y,$$

where $\oplus$ denotes an extended exclusive or.

Now the operating rules of M can be specified by a set R of sextuples $\langle \sigma_1, \ldots, \sigma_6 \rangle$, where σ_2 and one of $\sigma_4, \sigma_5, \sigma_6$ are in $K \times \Sigma$ and the other four components are in Σ. The significance of such a sextuple is that when M in the state indicated by σ_2 and its head is over the middle of three successive squares containing the symbols indicated by σ_1, σ_2, and σ_3, it may change its state, move its head, and rewrite the scanned square as indicated by σ_4, σ_5, σ_6. (These rules must be subjected to a slightly different interpretation to explicate the behavior of M when its head is over the leftmost tape square.) To stipulate that M operates in accordance with these rules we need the conjunction, for all $\sigma_1, \sigma_3 \in \Sigma$ and $\sigma_2 \in K \times \Sigma$, of

$$S_1 y \wedge S_2 y \wedge S_3 y \wedge S_4 y \wedge S_5 y \wedge P_{\sigma_1 1} y \wedge P_{\sigma_2 2} y \wedge P_{\sigma_3 3} y$$
$$\rightarrow \bigvee P_{\sigma_4 4} y \wedge P_{\sigma_5 5} y \wedge P_{\sigma_6 6} y,$$

the disjunction extending over all σ_4, σ_5, σ_6 such that $\langle \sigma_1, \ldots, \sigma_6 \rangle \in R$. (Some additional conjuncts, which we do not detail here, are needed to describe the operation of M when its head is over the leftmost tape square.) We also need to stipulate that tape squares not in the vicinity of the head do not change. For this the following clauses (plus some additional clauses to handle the leftmost and rightmost tape squares) are sufficient:

$$\bigwedge_{\sigma_1, \sigma_2, \sigma_3 \in \Sigma} (S_1 y \wedge S_2 y \wedge S_4 y \wedge P_{\sigma_1 1} y \wedge P_{\sigma_2 2} y \wedge P_{\sigma_3 3} y \rightarrow P_{\sigma_2 5} y).$$

It remains only to specify the values of $\mu(0, 0), \ldots, \mu(s-1, 0)$. In this we are aided by the fact that the pairs $(0, 0), \ldots, (s-1, 0)$ are the first components of he sextuples represented by the terms $\mathbf{0}, \ldots, \mathbf{s-1}$. Let us introduce $n+1$ new predicate letters N_i $(i = 0, \ldots, n-1)$ and Q, with these intended interpretations for argument $\mathbf{r}$:

$N_i\mathbf{r}$ is true if and only if $r = i$,

$Q\mathbf{r}$ is true if and only if $n \leqslant r \leqslant s-1$.

To ensure that $N_i\mathbf{r}$ or $Q\mathbf{r}$ is true at least under the specified circumstances (which is all that is actually required) we include the clauses

$$N_0 a,$$

$$\bigwedge_{i=0}^{n-2} (N_i y \to N_{i+1} f(y)),$$

$$N_{n-1} y \to Qf(y),$$

$$Qy \wedge \bigwedge_{i=0}^{11dn-1} \neg B_i f(y) \to Qf(y).$$

Now recall that $w = w_0 \cdots w_{n-1}$, each $w_i \in \Sigma$, and that **B** is the blank symbol. To complete the construction we need the clauses

$$N_0 y \to P_{(q_0, w_0)1} y,$$

$$\bigwedge_{i=1}^{n-1} (N_i y \to P_{w_i 1} y),$$

$$Qy \to P_{\mathbf{B}1} y.$$

The constructed formula is of length $O(n \log n)$ as required, when M is regarded as fixed. ∎

From this there follows by Proposition 2.1:

COROLLARY 4.2. *There is a constant $c > 1$ such that neither the Gödel nor the monadic class can be decided nondeterministically in time $c^{n/\log n}$.* ∎

5. UPPER BOUND: MONADIC CLASS

THEOREM 5.1. *For some $c > 0$,* Monadic–Sat $\in$ NTIME($c^{n/\log n}$).

Proof. We use a variant of the "miniscoping" method for deciding monadic formulas [27, p. 192]. This method, if applied ruthlessly, can result in a multiply exponential explosion in the size of the formula. Here we merge its application with a nondeterministic algorithm in such a way that as quantifiers are driven in to their minimum scopes, they are eliminated. The number of atomic formulas in the formula does not increase, so the length of the formula never grows beyond a single exponential in its original size.

It is known that a monadic formula F is satisfiable if and only if F has a model with at most 2^k elements, where k is the number of predicate letters of F. If n is the length of the formula, then $n = \Omega(k \log k)$, from which it follows that $k = O(n/\log n)$. Moreover, the model need have no two elements with the same *monadic index*, i.e., the same set of monadic predicates that are true of them.

The first step is to guess a model. This means writing down $p \leqslant 2^k$ bit vectors, one for each monadic index, each of length k.

Now we eliminate quantifiers one at a time. Choose some innermost quantifier, i.e., some quantifier with no other quantifier in its scope. Suppose it is a universal quantifier; a similar argument applies if it is existential. The quantifier and its scope have the form $\forall yG$, where G contains no quantifier. Put G into conjunctive normal form, say $G \equiv G_1 \wedge \cdots \wedge G_c$, where G_i is a disjunction of atomic formulas and negations of atomic formulas. Then

$$\forall yG \equiv \forall yG_1 \wedge \forall yG_2 \wedge \cdots \wedge \forall yG_c.$$

Next permute the G_i so that those in which y actually occurs are at the left; and permute the conjuncts of those G_i so that the atoms containing y are at the left of those disjunctions. Then drive the quantifier $\forall y$ inside the disjunctions, so that

$$\forall yG \equiv (\forall yG_1' \vee G_1'') \wedge (\forall yG_2' \vee G_2'') \wedge \cdots \wedge (\forall yG_d' \vee G_d'') \wedge G_{d+1}'' \wedge \cdots \wedge G_c''.$$

All the G_i' and G_i'' are disjunctions of signed atomic formulas. No G_i'' has any occurrence of y; each G_i' has occurrences of y but not of any other variable. Now refer to the model that was guessed, in order to determine whether $\forall yG_1', \ldots, \forall yG_d'$ are true or false; checking $\forall yG_i'$ simply involves seeing whether there is an element of the model that has the opposite value from that demanded by G_i' for each of the monadic predicates mentioned in G_i'. Replace each subformula $\forall yG_i'$ by "true" or "false" and simplify the result. This completes the elimination of one quantifier; repeat the same procedure for the other quantifiers, from innermost to outermost. (Existential quantifiers are eliminated by putting their scopes in *disjunctive* normal form.)

Suppose the original formula was of length n and contained m distinct atomic subformulas. Then $m = O(n/\log n)$. At any intermediate stage the longest formula that can be created is one containing 2^m disjuncts (or conjuncts), each of which is a conjunction (or a disjunction) of those original m formulas. Hence each conjunction (or disjunction) is of length $O(n/\log n)$, and the whole is of length $O((n/\log n) \cdot 2^{n/\log n})$. Since the number of quantifiers to be eliminated is bounded by n, the whole procedure takes nondeterministic time $c^{n/\log n}$ for some constant c. ∎

6. Upper Bound: Gödel Class

There is a constant $d > 0$ such that a formula in the Gödel class is satisfiable only if it has a model of cardinality at most d^m, where m is the number of predicate letters and variables (see [10]). Since formulas in this class are prenex with two universal quantifiers, Proposition 3.2 can be applied, with $p = d^m$ and $u = 2$. Since $m = O(|F|/\log |F|)$, it follows immediately that

Theorem 6.2. *For some* $c > 0$, $\exists^*\forall\forall\exists^*$–Sat $\in$ NTIME$(c^{n/\log n})$. ∎

7. Schönfinkel–Bernays Class: Lower Bound

THEOREM 7.1. NEXPTIME *is reducible to* $\exists^*\forall^*$–Sat *via length order n.*

Proof. As in Section 4, we reduce acceptance by nondeterministic exponential-time bounded Turing machines to formulas in the class. We are able to obtain a more economical reduction, however, by encoding numbers and strings over alphabets that grow in size as the length of the input increases.

Let S be a set in NTIME(c^n) for some c. For any $n \geqslant 0$, let $m(n)$ be the smallest integer such that $m(n)^{m(n)} \geqslant c^n$, or equivalently, such that $m(n) \log_c m(n) \geqslant n$. Thus $m(n) = O(n/\log n)$. Let M be a nondeterministic Turing machine that accepts S in time $m(n)^{m(n)}$. By an argument like that of Section 4, we may assume that M accepts an input of length n simply by having a computation on that input of $m(n)^{m(n)} - 1$ steps. Our construction yields, for any input string w to M, a formula F of length $O(|w|)$ that is satisfiable if and only M accepts w. (Again, the constant implicit in O-notation depends on M and c but not w.) We present the functional form of F, in which constants have replaced existentially quantified variables, as a conjunction of subformulas.

Let Σ be the tape alphabet of M and K the state set of M; let $\Gamma = \Sigma \cup (K \times \Sigma)$. Next, fix some input string w, and let $n = |w|$, $m = m(n)$, and $s = m^m$. As in Section 4, a computation by M of length s can be presented as a mapping μ: $\{0,..., s-1\} \times \{0,..., s-1\} \to \Gamma$.

Let Γ contain exactly g symbols and let $l = \lceil \log_g m \rceil$. We use as constants in our construction

(1) the members of Γ;

(2) m constants $d_0,..., d_{m-1}$, which represent the m digits in base m notation for integers;

(3) $g^{l+1} - 1 = O(m)$ constants to represent strings of symbols in Γ of length at most l; we denote these by $\langle\ \rangle$, $\langle\sigma\rangle$ (where $\sigma \in \Gamma$), $\langle\sigma_1\sigma_2\rangle$ (where $\sigma_1, \sigma_2 \in \Gamma$), and so on.

Our goal is to axiomatize a $(2m+1)$-place predicate letter P with the following intended interpretation:

> $Pt_0 \cdots t_{2m}$ is true if and only if $\langle t_0,..., t_{m-1}\rangle$ and $\langle t_m,..., t_{2m-1}\rangle$ represent a pair of numbers i, j such that $\mu(i,j) = t_{2m} \in \Gamma$.

Note that m-tuples of the constants $d_0,..., d_{m-1}$ are just sufficient to represent all numbers in the range $0,..., s-1$. It will follow that from any model for F an accepting computation of M on input w can be derived, and conversely, that from any such accepting computation a model for F can be specified.

As in the construction in Section 4, the conjuncts are of three kinds: formulas describing the successor relation, formulas describing the operation of M on any string of length n, and formulas specifying the particular input w. We begin with the

formulas describing the successor relation between numbers in the range $0,\ldots, s-1$. This will be represented by a $2m$-place predicate letter S with the intended interpretation that

$St_0 \cdots t_{2m-1}$ is true if and only if
the number represented by $\langle t_m,\ldots, t_{2m-1}\rangle$ is the successor of that represented by $\langle t_0,\ldots, t_{m-1}\rangle$.

Several auxiliary predicate letters are used in order to fix the interpretation of S, specifically, the dyadic predicate letters N, N^+, and E. Their intended interpretations are as follows:

$Nt_1 t_2$ is true if and only if,
for some $i = 0,\ldots, m-2$, $t_1 = d_i$ and $t_2 = d_{i+1}$.

$N^+ t_1 t_2$ is true if and only if,
for some i and j, $0 \leqslant i < j \leqslant m-1$, $t_1 = d_i$ and $t_2 = d_j$.

$Et_1 t_2$ is true if and only if,
for some $i = 0,\ldots, m-1$, $t_1 = t_2 = d_i$.

As axioms for these predicate letters we use the following formulas:

$$\bigwedge_{i=0}^{m-2} Nd_i d_{i+1},$$

$$Ny_1 y_2 \to N^+ y_1 y_2,$$

$$N^+ y_1 y_2 \wedge Ny_2 y_3 \to N^+ y_1 y_3 \wedge \neg Ny_1 y_3,$$

$$\neg Ny_1 y_1 \wedge (N^+ y_1 y_2 \to \neg Ny_2 y_1),$$

$$\bigwedge_{i=0}^{m-1} Ed_i d_i,$$

$$(N^+ y_1 y_2 \vee N^+ y_2 y_1) \to \neg Ey_1 y_2,$$

$$Ny_{m-1} y'_{m-1} \to \left(Sy_0 \cdots y_{m-1} y'_0 \cdots y'_{m-1} \leftrightarrow \bigwedge_{i=0}^{m-2} Ey_i y'_i\right),$$

$$Sy_0 \cdots y_{m-2} d_{m-1} y'_0 \cdots y'_{m-2} d_0$$
$$\leftrightarrow (Sd_0 y_0 \cdots y_{m-2} d_0 y'_0 \cdots y'_{m-2} \wedge \neg(Ey_0 d_{m-1} \wedge Ey'_0 d_0)),$$

$$\neg Ny_{m-1} y'_{m-1} \wedge \neg(Ey_{m-1} d_{m-1} \wedge Ey'_{m-1} d_0) \to \neg Sy_0 \cdots y_{m-1} y'_0 \cdots y'_{m-1}.$$

These clauses uniquely determine the interpretations of N, N^+, and S on all systems of arguments from among $d_0,\ldots, d_{m-1}$. (Their interpretations on other arguments are not important; in the construction of a verifying truth-assignment, any other constant could here be regarded as indistinguishable from d_0.) Note in particular that the last

three formulas specify the conditions under which two m-tuples $\langle t_0,..., t_{m-1}\rangle$ and $\langle t'_0,..., t'_{m-1}\rangle$ stand in the successor relation: (1) if t_{m-1}, t'_{m-1} are a successive pair of digits d_i, d_{i+1}, then the m-tuples are successive if and only if the first $m-1$ digits are pairwise identical; (2) if $t_{m-1} = d_{m-1}$ and $t'_{m-1} = d_0$, then the m-tuples are successive if and only if their first $m-1$ digits, prefixed by d_0, are in the successor relation, and their first digits are not d_{m-1} and d_0, respectively; and (3) otherwise, the two m-tuples are not successive.

We can now turn to the subformulas describing the operation of M. We must first stipulate that μ is a function:

$$\bigoplus_{\sigma\in\Gamma} Py_0 \cdots y_{2m-1}\sigma.$$

As in Section 4, the operating rules of M can be specified by a set R of sextuples $\langle\sigma_1,..., \sigma_6\rangle$. Then we include as conjuncts all of the following formulas, where $\sigma_1, \sigma_3 \in \Sigma$ and $\sigma_2 \in K \times \Sigma$:

$$\begin{aligned}
&(Px_0 \cdots x_{m-1} y_0 \cdots y_{m-1}\sigma_1\\
&\quad \wedge Px'_0 \cdots x'_{m-1} y_0 \cdots y_{m-1}\sigma_2\\
&\quad \wedge Px''_0 \cdots x''_{m-1} y_0 \cdots y_{m-1}\sigma_3\\
&\quad \wedge Sx_0 \cdots x_{m-1} x'_0 \cdots x'_{m-1}\\
&\quad \wedge Sx'_0 \cdots x'_{m-1} x''_0 \cdots x''_{m-1}\\
&\quad \wedge Sy_0 \cdots y_{m-1} y'_0 \cdots y'_{m-1})\\
&\rightarrow \bigvee (Px_0 \cdots x_{m-1} y'_0 \cdots y'_{m-1}\sigma_4\\
&\quad \wedge Px'_0 \cdots x'_{m-1} y'_0 \cdots y'_{m-1}\sigma_5\\
&\quad \wedge Px''_0 \cdots x''_{m-1} y'_0 \cdots y'_{m-1}\sigma_6),
\end{aligned}$$

the disjunction extending over all $\langle\sigma_4, \sigma_5, \sigma_6\rangle$ such that $\langle\sigma_1,..., \sigma_6\rangle \in R$. (Again, some additional conjuncts, which we do not detail, are needed to describe the operation of M when its head is over leftmost tape square.) We also need to stipulate that tape squares not in the vicinity of the head do not change. For this we can write

$$\begin{aligned}
\bigwedge_{\sigma_1,\sigma_2,\sigma_3\in\Sigma} &(Px_0 \cdots x_{m-1} y_0 \cdots y_{m-1}\sigma_1\\
&\quad \wedge Px'_0 \cdots x'_{m-1} y_0 \cdots y_{m-1}\sigma_2\\
&\quad \wedge Px''_0 \cdots x''_{m-1} y_0 \cdots y_{m-1}\sigma_3\\
&\quad \wedge Sx_0 \cdots x_{m-1} x'_0 \cdots x'_{m-1}\\
&\quad \wedge Sx'_0 \cdots x'_{m-1} x''_0 \cdots x''_{m-1}\\
&\quad \wedge Sy_0 \cdots y_{m-1} y'_0 \cdots y'_{m-1}\\
&\rightarrow Px'_0 \cdots x'_{m-1} y'_0 \cdots y'_{m-1}\sigma_2),
\end{aligned}$$

plus additional similar conjuncts applying to the leftmost and rightmost tape squares.

It remains only to ensure that the values of $\mu(0,0),\ldots,\mu(s-1,0)$ are correct, namely, that $\mu(0,0)=(q_0,w_0)$, where q_0 is the initial state and w_0 is the first symbol of w; $\mu(i,0)=w_i$ for $i=1,\ldots,n-1$, where w_i is the $(i+1)$st symbol of w; and $\mu(i,0)=\mathbf{B}$, the blank symbol, for $n\leqslant i\leqslant s-1$. In order to specify $\mu(0,0),\ldots,\mu(n-1,0)$ succinctly we shall use a sequence of $p=\lceil n/l\rceil$ constants, each representing a substring of w of length l at most. We shall also need formulas for unpacking these blocks to obtain their individual symbols. Note that

$$p=\left\lceil\frac{n}{l}\right\rceil=\left\lceil\frac{n}{\lceil\log_g m\rceil}\right\rceil=O(m).$$

We use a triadic predicate letter Q to represent a $3\times(g^{l+1}-2)$ table of the strings having length between 1 and l inclusive, the first symbol of each, and the string that results when the first symbol is removed. Let $\rho_1,\sigma_1,\phi_1,\ldots,\rho_\lambda,\sigma_\lambda,\phi_\lambda$ be a sequence of all such triples; that is, $\lambda=g^{l+1}-2$, $\{\rho_1,\ldots,\rho_\lambda\}=\{u\colon u\in\Gamma^*$ and $1\leqslant|x|\leqslant l\}$, $\sigma_1,\ldots,\sigma_\lambda\in\Gamma$, and $\rho_i=\sigma_i\phi_i$ for each i. Then we include as a conjunct the formula

$$\bigwedge_{i=1}^{\lambda} Q\langle\rho_i\rangle\,\sigma_i\langle\phi_i\rangle.$$

We need one further predicate I, which has $m+p$ argument-places and the following intended interpretation:

$It_0\cdots t_{m-1}u_1\cdots u_p$ is true if and only if
$\langle t_0,\ldots,t_{m-1}\rangle$ represents a number i and $\langle u_1,\ldots,u_p\rangle$ represents, as a sequence of substrings, the values of $\mu(i,0),\mu(i+1,0),\ldots,\mu(s-1,0)$.

Of course a sequence of only n symbols in Γ can be represented explicitly by $\langle u_1,\ldots,u_p\rangle$, but this suffices since $\mu(n,0),\ldots,\mu(s-1,0)$ must all be the blank symbol $\mathbf{B}$. We use the relation Q to unpack u_1; when u_1 has been reduced to the constant $\langle\ \rangle$ denoting the empty string the last $p-1$ arguments of I are shifted left one, the rightmost argument being replaced by a special constant a. (This could be d_0 or any other constant not being used to represent a string of symbols in Γ.) When all the string-type arguments of I have been unpacked an occurrence of a reaches the u_1 position, and thereafter the values of $\mu(i,0)$ are required to be blanks.

To be pecise, let $\psi_0,\ldots,\psi_{p-1}$ be the l-fold compression of $\mu(0,0),\ldots,\mu(n-1,0)$; that is, ψ_i is the string $\mu(li,0)\,\mu(li+1,0)\cdots\mu(li+l-1,0)$. Then we include as conjuncts the following formulas:

$$Id_0\cdots d_0\langle\psi_0\rangle\cdots\langle\psi_{p-1}\rangle$$

$$Ix_0\cdots x_{m-1}z_0\cdots z_{p-1}\wedge Qz_0yz\to Px_0\cdots x_{m-1}d_0\cdots d_0y$$
$$\wedge(Sx_0\cdots x_{m-1}x_0'\cdots x_{m-1}'\to Ix_0'\cdots x_{m-1}'zz_1\cdots z_{p-1}),$$

$$Ix_0\cdots x_{m-1}\langle\ \rangle z_1\cdots z_{p-1}\to Ix_0\cdots x_{m-1}z_1\cdots z_{p-1}a,$$

$$Ix_0\cdots x_{m-1}az_1\cdots z_{p-1}\to Px_0\cdots x_{m-1}d_0\cdots d_0\mathbf{B},$$
$$\wedge(Sx_0\cdots x_{m-1}x_0'\cdots x_{m-1}'\to Ix_0'\cdots x_{m-1}'az_1\cdots z_{p-1}).$$

This completes the construction, the required formula is the conjunction of all the subformulas given above. Each of these formulas has length $O(m \log m) = O(n)$ as required. ∎

COROLLARY. *There is a* $c > 1$ *such that* $\exists^*\forall^*$–Sat *is not decidable nondeterministically in time* c^n. ∎

8. SCHÖNFINKEL–BERNAYS CLASS: UPPER BOUND

To obtain an upper bound for the complexity of $\exists^*\forall^*$–Sat, we apply the general analysis of Section 3, specifically Proposition 3.1, that the existence of a model of cardinality p for a formula F with q quantifiers can be checked nondeterministically in time $c^{q \log p + \log |F|}$. A formula $F = \exists x_1 \cdots x_k \forall y_1 \cdots y_m F_0$, where F_0 is quantifier-free, has a model only if it has a model of size at most k. By the argument at the end of Section 2, it follows that $k = O(|F|/\log |F|)$ (if the x_i are to be distinct) and $q = k + m = O(|F|/\log |F|)$. Substituting these values yields:

THEOREM 8.1. $\exists^*\forall^*$–Sat $\in$ NEXPTIME. ∎

9. ACKERMANN CLASS: LOWER BOUND

This is the prefix class $\exists^*\forall\exists^*$.

THEOREM 9.1. DEXPTIME *is reducible to* $\exists^*\forall\exists^*$–Sat *via length order* $n \log n$.

Proof. We reduce to the satisfiability problem for the.class the problem of whether an alternating pushdown automaton does *not* accept an input string; by the results of Section 2C this is a reduction of DEXPTIME to $\exists^*\forall\exists^*$–Sat. Specifically, let $S \subseteq \Sigma^*$ be any set in DEXPTIME. By Proposition 2.4, $\Sigma^* - S$ is also in DEXPTIME, and by Proposition 2.2 there is an alternating pushdown automaton that accepts $\Sigma^* - S$. Let M be such an alternating pushdown automaton, and let K, Σ, Γ, $\vdash$, $\dashv$, Z, s, f, $K_1, \ldots, K_7$, and $\delta_1, \ldots, \delta_7$ be for M as described in Section 2C. Let u be any string in Σ^*, and let $w = \vdash u \dashv$. We construct a formula G which is the functional form of a formula F in $\exists^*\forall\exists^*$; F is satisfiable if and only if M does not accept u. This amounts to a reduction of S to $\exists^*\forall\exists^*$–Sat. Moreover, the length of F will be $O(n \log n)$, where $n = |u|$, so the theorem will be proved.

For each state $q \in K$ and each i, $1 \leqslant i \leqslant |w|$, we introduce a distinct monadic predicate letter. (Here i is one of the possible positions of the reading head.) To avoid multiple subscripts, we write Q_i for the predicate letter corresponding to state q and head position i; thus F_i is that corresponding to the final state f and head position i, etc. We also introduce a monadic function sign g_X for each symbol $X \in \Gamma - \{Z\}$, and a constant sign c_Z corresponding to the bottom marker Z. Thus each term in the

Herbrand universe is of the form $g_{X_1}(g_{X_2}(\cdots(g_{X_n}(c_Z))\cdots))$ for some $X_1,\ldots,X_n \in \Gamma - \{Z\}$; this term represents the stack $X_1 \cdots X_n Z$. In general, we write γ for the term in the Herbrand universe that represents the stack $\gamma \in (\Gamma - \{Z\})^* Z$. The intended interpretation of an atomic formula $Q_i\gamma$ is that the configuration (q, w, i, γ) is accepting.

The formula G has one variable y and is the conjunction of three parts G_1, G_2, G_3. G_1 states that certain configurations are accepting; G_2 states that the initial configuration is not accepting; and G_3 states that if certain configurations are accepting then so are certain others.

G_1 is $\bigwedge_{i=1}^{|w|} F_i y$ (where F_i is the predicate letter corresponding to state f and head position i).

G_2 is $\neg S_1 c_Z$ (where S_1 is the predicate letter corresponding to the start state s and head position 1).

G_3 is the conjunction of all the subformulas obtained from the states of M as follows:

Read states. Let $q \in K_1$, and let $1 \leqslant i \leqslant |w|$. Let a be the ith symbol of w, and let $\delta_1(q, a) = p$. Then G_3 has the conjunct

$$P_i y \to Q_i y.$$

Move states. Let $q \in K_2$, and let $1 \leqslant i \leqslant w$. Let $\delta_2(q) = (p, \varepsilon)$. Then if $1 \leqslant i + \varepsilon \leqslant |w|$, G_3 has the conjunct

$$P_{i+\varepsilon} y \to Q_i y.$$

Push states. Let $q \in K_3$, and let $1 \leqslant i \leqslant |w|$. Let $\delta_3(q) = (p, X)$. Then G_3 has the conjunct

$$P_i g_X(y) \to Q_i y.$$

Pop states. Let $q \in K_4$, and let $1 \leqslant i \leqslant |w|$. Let $\delta_4(q) = p$. Then for each $X \in \Gamma - \{Z\}$, G_3 has the conjunct

$$P_i y \to Q_i g_X(y).$$

Pushdown store read states. Let $q \in K_5$, let $1 \leqslant i \leqslant |w|$. Let $X \in \Gamma$, and let $\delta_5(q, X) = p$. Then G_3 has the conjunct

$$P_i g_X(y) \to Q_i g_X(y) \quad \text{if} \quad X \text{ is not } Z,$$
$$P_i c_Z \to Q_i c_Z \qquad \text{if} \quad X \text{ is } Z.$$

Existential branch states. Let $q \in K_6$ and let $1 \leqslant i \leqslant |w|$. Let $\delta_6(q) = \{p,\ldots, r\}$. Then G_3 has the conjunct

$$(P_i y \vee \cdots \vee R_i y) \to Q_i y.$$

Universal branch states. Let $q \in K_7$ and let $1 \leqslant i \leqslant |w|$. Let $\delta_7(q) = \{p,..., r\}$. Then G_3 has the conjunct

$$(P_i y \wedge \cdots \wedge R_i y) \rightarrow Q_i y.$$

This completes the construction of G. We claim:

(1) If $\mathscr{A}$ is any truth-assignment verifying each instance of G_1 and each instance of G_3, then for any $\gamma \in (\Gamma - \{Z\})^* Z$, $\mathscr{A}$ verifies $Q_i\gamma$ if (q, w, i, γ) is an accepting configuration of M.

(2) The following truth-assignment $\mathscr{A}$ verifies each instance of G_1 and each instance of G_3: $\mathscr{A}$ verifies an atomic formula $Q_i\gamma$, where $\gamma \in (\Gamma - \{Z\})^* Z$, if and only if (q, w, i, γ) is an accepting configuration of M.

Before proceeding, we note that (1) and (2) are sufficient to establish that F is satisfiable if and only if M does not accept u. For if F is satisfiable, then there is a truth-assignment $\mathscr{A}$ verifying each instance of G_1, each of G_2, and each of G_3. By (1), $\mathscr{A}$ must verify $Q_i\gamma$ xhenever (q, w, i, γ) is an accepting configuration of M. But $\mathscr{A}$ must also verify $\neg S_1 \mathbf{Z}$, which is the sole instance of G_2. Therefore $(s, w, 1, Z)$ is not an accepting configuration. But this is the initial configuration, so M does not accept u. Conversely, if M does not accept u, then the truth-assignment $\mathscr{A}$ of (2) verifies not only the instances of G_1 and of G_3, but also the instance of G_2, since then $(s, w, 1, Z)$ is not an accepting configuration.

(1) is proved by induction. If (q, w, i, γ) is accepting because q is the final state f, then $\mathscr{A}$ must verify $Q_i\gamma$ since then $Q_i\gamma$ is an instance of one of the conjuncts of G_1. Otherwise a case-analysis must be carried out, depending on the reason (q, w, i, γ) is accepting. We argue just one case, the rest being similar. Suppose that q is a push state, and $\delta_3(q) = (p, X)$. Then (q, w, i, γ) is accepting because $(p, w, i, X\gamma)$ is accepting. If we assume by induction that (1) holds for the latter configuration, it follows that $\mathscr{A}$ verifies $P_i X\gamma$, that is, $P_i g_X(\gamma)$. But $P_i g_X(\gamma) \rightarrow Q_i\gamma$ is an instance of a conjunct of G_3 (in which γ has been substituted for y), so if $\mathscr{A}$ verifies each instance of G_3 then $\mathscr{A}$ must verify $Q_i\gamma$ as well.

The proof of (2) also requires a case-analysis. The truth-assignment $\mathscr{A}$ defined in (2) verifies each instance of G_1, since each such instance is a conjunction of atomic formulas $F_i\gamma$, and (f, w, i, γ) is always accepting. As for the instances of G_3 we again argue just one case, the rest being similar. Consider a conjunct $P_i g_X(\gamma) \rightarrow Q_i\gamma$, where q is a push state, and suppose $\mathscr{A}$ verifies $P_i g_X(\gamma)$ (otherwise this conjunct is verified automatically, since the antecedent is false). By the definition of $\mathscr{A}$, $(p, w, i, X\gamma)$ is accepting. But then (q, w, i, γ) is accepting, since $\delta_3(q) = (p, X)$. Therefore by the definition of $\mathscr{A}$ again $\mathscr{A}$ verifies $Q_i\gamma$, so $\mathscr{A}$ verifies the conjunct $P_i g_X(\gamma) \rightarrow Q_i\gamma$ as was to be shown.

Finally, note that if M is regarded as fixed then G has $O(|w|)$ predicate letters and is the conjunction of $O(|w|)$ subformulas. The reduction is therefore via length order $n \log n$ as claimed. ∎

From this theorem there follows immediately by Proposition 2.1:

COROLLARY 9.2. *There is a* $c > 1$ *such that* $\exists^*\forall\exists^*$–Sat *is not decidable deterministically in time* $c^{n/\log n}$. ∎

10. ACKERMANN CLASS: UPPER BOUNDS

In this section we establish an upper bound for the monadic Ackermann class comparable to the lower bound of Corollary 9.2. Before doing so, however, we introduce some terminology and notation useful both here and in Section 12.

A *signed atomic formula* is an atomic formula or the negative of one; two signed atomic formulas are *cosigned* if they are both atomic formulas or are both negations of atomic formulas. If D_1 and D_2 are conjunctions of signed atomic formulas, then we write $D_1 \sim D_2$ if and only if any atomic formula appearing in both D_1 and D_2 appears in cosigned conjuncts of D_1 and D_2. For example, if A_1, A_2, and A_3 are distinct atomic formulas, then

$$A_1 \wedge \neg A_2 \sim \neg A_2 \wedge A_3$$

but

$$A_1 \wedge A_3 \not\sim \neg A_3.$$

(Note that $\sim$ is *not* an equivalence relation.)

For any conjunction D of signed atomic formulas and for any variables $v_1,\ldots, v_n$, let $D|v_1,\ldots, v_n$ be the subconjunction of D consisting of those signed atomic formulas whose variables are among $v_1,\ldots, v_n$.

A disjunctive normal form of a formula F is said to be *full* if it consists of a disjunction of conjunctions of signed atomic formulas such that every disjunct contains an occurrence of every atomic formula that appears in F. Thus a full disjunctive normal form can be arranged so that all disjuncts look alike except for the placement of negation signs.

THEOREM 10.1. *For some* $c > 0$, $\exists^*\forall\exists^* \cap$ Monadic–Sat $\in$ DTIME$(c^{n/\log n})$.

Proof. Let $F = \exists z_1 \cdots \exists z_k \forall y \exists x_1 \cdots \exists x_m F_0$, where F_0 is quantifier-free. We claim that to determine whether F is satisfiable it suffices to apply the following test.

Test (Kalmár–Schütte). 1. Put F_0 into full disjunctive normal form, say G_0.

2. Check to see that there is a subset S of the disjuncts of G_0 such that

(a) For every pair D_1, D_2 of disjuncts in S,

$$D_1 \mid z_1,\ldots, z_k \sim D_2 \mid z_1,\ldots, z_k.$$

(b) For each variable z_j there is a disjunct D_1 in S such that for every disjunct D_2 in S,

$$(D_1 \mid z_1,\ldots, z_k, y)\mid y/z_j \mid \sim D_2 \mid z_1,\ldots, z_k.$$

(c) For each disjunct D_1 in S and each variable x_j, there is a disjunct D_2 in S such that

$$D_1 \mid z_1,\ldots, z_k, x_j \sim (D_2 \mid z_1,\ldots, z_k, y)[y/x_j].$$

The test is correct. For let a_j be the constant associated with z_j and f_j the monadic function sign associated with x_j in the functional form of F. Thus every term in the Herbrand universe $D(F)$ is of the form $f_{j_1}(f_{j_2}(\cdots (f_{j_n}(a_i))\cdots))$ for some $n \geqslant 0, j_1,\ldots,j_n$, and i.

First suppose that there is a subset S of the disjuncts of G_0 satisfying (a), (b), and (c). Then define a mapping $\mu: D(F) \to S$ thus:

(1) For each constant sign a_j, $\mu(a_j)$ is some disjunct D_1 as specified in (b) for the variable z_j;

(2) for each term $s = f_j(t)$ that is not a constant sign, $\mu(s)$ is some D_2 as specified in (c) for the disjunct $D_1 = \mu(t)$ and the variable x_j.

Then let $\mathscr{A}$ be a truth-assignment to the Herbrand expansion such that $\mathscr{A}$ verifies an atomic formula A if and only if A is a conjunct of $\mu(t)^*[y/t]$, where $\mu(t)^*$ is the formula that supplants disjunct $\mu(t)$ in the functional form of F. We claim that $\mathscr{A}$ verifies each instance of the functional form of F, specifically, that $\mathscr{A}$ verifies disjunct $\mu(t)^*[y/t]$ in that instance of the functional form in which t is the substituent for y. This could fail to be the case only if $\mu(t)^*[y/t]$ had some negated atomic formula $\neg A$ as a conjunct such that A was a conjunct of $\mu(s)^*[y/s]$ for some other term s. But this is impossible, since A can appear in both $\mu(t)^*[y/t]$ and $\mu(s)^*[y/s]$ only under circumstances in which (a), (b), (c), and the definition of μ guarantee that the appearances would be cosigned. Clauses (a) and (b) take care of the case in which A contains only constants; (a) the case in which A itself appears in $\mu(t)^*$ and $\mu(s)^*$, and (b) the case in which A appears in one of $\mu(t)^*$ and $\mu(s)^*$ and emerges in the other when a constant sign is substituted for y. And clause (c) takes care of the case in which A contains a non-constant term; here A can appear in both $\mu(t)^*[y/t]$ and $\mu(s)^*[y/s]$ $(t \neq s)$ only if $s = f_j(t)$ for some j (or vice versa) and, for some atomic formulas A_1 in $\mu(t)^*$ and A_2 in $\mu(s)^*$, $A_1[y/t] = A_2[y/s]$. But then A_1 and A_2 must be identical, except that A_1 has an occurrence of $f_j(y)$ where A_2 has an occurrence of y, and then (c) guarantees that the signed atomic formulas in $\mu(t)^*$ and $\mu(s)^*$ in which A_1 and A_2 (respectively) appear are cosigned.

The test can be carried out in time $O(c^{n/\log n})$ for some constant $c > 0$, where n is the length of F. The full disjunctive normal form G_0 of F_0 has length $O(c_1^{n/\log n})$ for some $c_1 > 0$. A set S of disjuncts of G_0 satisfying (a), (b), and (c) can be found (provided one exists) as follows:

(i) (Ensure that (a) holds.) Partition the disjuncts of G_0 into maximal subsets such that for each pair of disjuncts D_1, D_2 in the same subset,

$$D_1 \mid z_1,\ldots, z_k \sim D_2 \mid z_1,\ldots, z_k.$$

Although $\sim$ is not an equivalence relation in general, the partitioning can be done in this case because the disjunctive normal form is full. Now carry out steps (ii) and (iii) for each member-set S' of the partition.

(ii) (Ensure that (c) holds.) Repeat this "crossing-out" operation until no new disjuncts are eliminated from S': for each disjunct D_1 that has not already been crossed out and for each x_j, if no satisfactory D_2 exists among the disjuncts in S' that have not been crossed out, then cross out D_1.

(iii) (Ensure that (b) holds.) Scan the remaining disjuncts in S' once for each variable z_j to see that there is a D_1 such that for every D_2 in S',

$$(D_1 \mid z_1,..., z_k, y)[y/z_j] \sim D_2 \mid z_1,..., z_k.$$

(Because $D \mid z_1,..., z_k$ is the same for all disjuncts D in S', it is enough to check this condition for any one D_2 in S'—for example, D_1.)

Now (ii) requires at most as many repetitions as there are disjuncts in S' to cross out; each repetition requires scanning G_0 once for each remaining disjunct D_1 and each variable x_j. Thus (ii) can be done with $|S'|^2 m$ scans of G_0 for a total time of $O(|G_0|^4)$. Similarly (iii) can be done in time polynomial in the length of G_0; and since the total number of member-sets S' in the partition found in (i) is $O(|G_0|)$, the whole test takes time polynomial in $|G_0|$ and therefore $O(c^{n/\log n})$ for some $c > 0$.

This completes the proof of Theorem 10.1. ∎

For Ackermann formulas with polyadic predicate letters, step (b) must be replaced by a more complicated test, and an upper bound of DTIME$(c^{(n/\log n)^2})$ results. (See the Acknowledgments.)

11. The Class $\mathscr{F}$: Lower Bound

The class $\mathscr{F}$ is defined in Section 2A.

THEOREM 11.1. D2EXPTIME *is reducible to* $\mathscr{F}$–Sat *via length order* $n \log n$.

Proof. The proof of this theorem closely follows that of Theorem 9.1, except that the automata used are alternating stack automata rather than alternating pushdown automata.

Let $S \subseteq \Sigma^*$ be a set in D2EXPTIME. By Propositions 2.3 and 2.4, there is an alternating stack automaton M that accepts $\Sigma^* - S$. Let K, Σ, Γ, $\vdash$, $\dashv$, Z, s, f, $K_1,..., K_9$, and $\delta_1,..., \delta_9$ be for M as described in Section 2C. Let u be any string in Σ^*. We construct the functional form G of a formula F in $\mathscr{F}$ which is satisfiable if and only if M does not accept u; this amounts to a reduction of S to $\mathscr{F}$–Sat. Moreover $|F| = O(n \log n)$, where $n = |u|$, so the theorem will be proved.

Let $w = \vdash u \dashv$. For each $q \in K$ and i, $1 \leqslant i \leqslant |w|$, we introduce a dyadic predicate letter Q_i in the way we introduced monadic predicate letters in Section 10. Likewise, let g_X be a monadic function sign for each $X \in \Gamma$ and let c_Z be a constant sign. We

adopt the same encoding of strings as terms: If $\gamma = X_1 \cdots X_n Z$, then $\boldsymbol{\gamma} = g_{X_1}(g_{X_2}(\cdots (g_{X_n}(c_Z)) \cdots))$. However, an atomic formula $Q_i \boldsymbol{\gamma}_1 \boldsymbol{\gamma}_2$ is now to be interpreted as true if and only if γ_1 is a suffix of γ_2 (i.e., $\gamma_2 = \gamma_3 \gamma_1$ for some $\gamma_3 \in (\Gamma - \{Z\})^*$) and $(q, w, i, \gamma_2, |\gamma_1|)$ is an accepting configuration of M. That is, the second argument represents the stack, and the first argument represents that portion of the stack from the bottom-marker up to and including the position of the stack head. To enforce this interpretation of the predicate letters Q_i we must introduce one further dyadic predicate letter Π; $\Pi \boldsymbol{\gamma}_1 \boldsymbol{\gamma}_2$ is to be true if and only if γ_1 is a suffix of γ_2.

The variables of G are y_1 and y_2. G is the conjunction of four subformulas $G_0, \ldots, G_3$.

G_0 is

$$\Pi y_1 y_2 \wedge \bigwedge_{X \in \Gamma - \{Z\}} (\Pi g_X(y_1) y_2 \to \Pi y_1 y_2).$$

G_1 and G_2 are similar to the formulas of the same name in Section 9. G_1 is

$$\Pi y_1 y_2 \to \bigwedge_{i=1}^{|w|} F_i y_1 y_2$$

and G_2 is

$$\neg S_1 c_Z c_Z.$$

G_3 is a conjunction of nine parts, the first seven of which are similar to subformulas constructed in Section 9, except that we must now capture the idea that in states of certain types the machine may move even when its head is in the interior of the stack.

Read states. Let $q \in K_1$, and let $1 \leqslant i \leqslant |w|$. Let a be the ith symbol of w, and let $\delta_1(q, a) = p$. Then G_3 has the conjunct

$$P_i y_1 y_2 \to Q_i y_1 y_2.$$

Move states. Let $q \in K_2$, and let $1 \leqslant i \leqslant |w|$. Let $\delta_2(q) = (p, \varepsilon)$. Then if $1 \leqslant i + \varepsilon \leqslant |w|$, G_3 has the conjunct

$$P_{i+\varepsilon} y_1 y_2 \to Q_i y_1 y_2.$$

Push states. Let $q \in K_3$, and let $1 \leqslant i \leqslant |w|$. Let $\delta_3(q) = (p, x)$. Then G_3 has the conjunct

$$P_i g_X(y_1) g_X(y_1) \to Q_i y_1 y_1.$$

Pop states. Let $q \in K_4$, and let $1 \leqslant i \leqslant |w|$. Let $\delta_4(q) = p$. Then for each $X \in \Gamma - \{Z\}$, G_3 has the conjunct

$$P_i y_1 y_1 \to Q_i g_X(y_1) g_X(y_1).$$

Stack read states. Let $q \in K_5$, and let $1 \leqslant i \leqslant |w|$. Let $X \in \Gamma$, and let $\delta_5(q, x) = p$. Then G_3 has the conjunct

$$P_i g_X(y_1) y_2 \to Q_i g_X(y_1) y_2 \qquad \text{if} \quad X \neq Z$$
$$P_i c_Z y_2 \to Q_i c_Z y_2 \qquad \text{if} \quad X = Z.$$

Existential branch states. Let $q \in K_6$ and let $1 \leqslant i \leqslant |w|$. Let $\delta_6(q) = \{p, \ldots, r\}$. Then G_3 has the conjunct

$$(P_i y_1 y_2 \vee \cdots \vee R_i y_1 y_2) \to Q_i y_1 y_2.$$

Universal branch states. Let $q \in K_7$ and let $1 \leqslant i \leqslant |w|$. Let $\delta_7(q) = \{p, \ldots, r\}$. Then G_3 has the conjunct

$$(P_i y_1 y_2 \wedge \cdots \wedge R_i y_1 y_2) \to Q_i y_1 y_2.$$

Stack head down states. Let $q \in K_8$, and let $1 \leqslant i \leqslant |w|$. Let $\delta_8(q) = p$. Then for every $X \in \Gamma - \{Z\}$, G_3 has the conjunct

$$P_i y_1 y_2 \wedge \Pi g_X(y_1) y_2 \to Q_i g_X(y_1) y_2.$$

Stack head up states. Let $q \in K_9$, and let $1 \leqslant i \leqslant |w|$. Let $\delta_9(q) = p$. Then for every $X \in \Gamma - \{Z\}$, G_3 has conjunct

$$P_i g_X(y_1) y_2 \to Q_i y_1 y_2.$$

This completes the construction of G. We claim

(1) If $\mathscr{A}$ is any truth-assignment verifying each instance of G_0, each instance of G_1, and each instance of G_3, then for any $\gamma_1, \gamma_2 \in (\Gamma - \{Z\})^*$ such that γ_1 is a suffix of γ_2, $\mathscr{A}$ verifies $Q_i \gamma_1 \gamma_2$ if $(q, w, i, \gamma_2, |\gamma_1|)$ is an accepting configuration of M.

(2) The following truth-assignment $\mathscr{A}$ verifies each instance of G_0, each instance of G_1, and each instance of G_3: $\mathscr{A}$ verifies $\Pi \gamma_1 \gamma_2$ if and only if γ_1 is a suffix of γ_2; and $\mathscr{A}$ verifies $Q_i \gamma_1 \gamma_2$ if and only if γ_1 is a suffix of γ_2 and $(q, w, i, \gamma_2, |\gamma_1|)$ is an accepting configuration of M.

As in the proof of Theorem 9.1, it follows readily from (1) and (2) that F is satisfiable if and only if M does not accept u, since G_2 may be interpreted as asserting that the initial configuration is not an accepting configuration.

To prove (1), first note that any truth-assignment verifying each instance of G_0 must verify $\Pi \gamma_1 \gamma_2$ whenever γ_1 is a suffix of γ_2. By inspection of G_1, then, it follows that any truth-assignment verifying each instance of G_0 and each instance of G_1 must verify $F_i \gamma_1 \gamma_2$ whenever $1 \leqslant i \leqslant |w|$ and γ_1 is a suffix of γ_2. To show that a truth-assignment $\mathscr{A}$ that verifies each instance of G_0, each of G_1, and each of G_3 also verifies $Q_i \gamma_1 \gamma_2$ when γ_1 is a suffix of γ_2 and $(q, w, i, \gamma_2, |\gamma_1|)$ is an accepting configuration, we must again argue by induction, depending on the configuration or configurations that follow $(q, w, i, \gamma_2, |\gamma_2|)$. As in Section 9, we take just one case, this

time the case in which q is a stack head down state. Let $\gamma_1 = X\gamma_3$ for some $X \in \Gamma - \{Z\}$, $\gamma_3 \in (\Gamma - \{Z\})^* Z$. (If $\gamma_1 = Z$ then $(q, w, i, \gamma_2, |\gamma_1|)$ cannot be accepting, unless q is f, a case already taken care of.) Let $\delta_8(q) = p$. Then $(q, w, i, \gamma_2, |\gamma_1|)$ is accepting provided that $(p, w, i, \gamma_2, |\gamma_1| - 1)$ is accepting, where $|\gamma_1| - 1 = |\gamma_3|$. Assume by induction that $\mathscr{A}$ verifies $P_i\gamma_3\gamma_2$, and note that one of the conjuncts of G_3 has an instance

$$P_i\gamma_3\gamma_2 \wedge \Pi g_X(\gamma_3)\,\gamma_2 \to Q_i\, g_X(\gamma_3)\,\gamma_2$$

obtained by substituting γ_3 for y_1 and γ_2 for y_2. Since $X\gamma_3$ is a suffix of γ_2, $\mathscr{A}$ must verify $\Pi g_X(\gamma_3)\,\gamma_2$ as well as $P_i\gamma_3\gamma_2$, so $\mathscr{A}$ verifies $Q_i\, g_X(\gamma_3)\,\gamma_2$. But this is $Q_i\gamma_1\gamma_2$.

To prove (2), we again consider just the case of a stack head down state q. Let $\delta_8(q) = p$. Let $\gamma_3, \gamma_2 \in (\Gamma - \{Z\})^* Z$, and consider the instance

$$P_i\gamma_3\gamma_2 \wedge \Pi g_X(\gamma_3)\,\gamma_2 \to Q_i\, g_X(\gamma_3)\,\gamma_2.$$

Suppose that both the antecedents are verified by $\mathscr{A}$. Then $(p, w, i, \gamma_2, |\gamma_3|)$ is an accepting configuration, and $X\gamma_3$ is a suffix of γ_2. But then $(q, w, i, \gamma_2, |\gamma_3| + 1)$ is an accepting configuration so $\mathscr{A}$ must also verify $Q_i\, g_X(\gamma_3)\,\gamma_2$.

Finally, we note as before that if M is regarded as fixed then G and hence F is of length $O(n \log n)$, where $n = |u|$, since the number of predicate letters and subformulas is linear in $|w|$. This completes the proof of Theorem 11.1.

From Theorem 11.1 there follows by Proposition 2.1:

COROLLARY 11.2. *There is a $c > 1$ such that $\mathscr{F}$–Sat is not decidable deterministically in time $c^{cn/\log n}$.* ∎

12. THE CLASS $\mathscr{F}$: UPPER BOUND

THEOREM 12.1. *For some $c > 0$, $\mathscr{F}$–Sat $\in$ DTIME($c^{cn/\log n}$).*

Proof. We reduce $\mathscr{F}$ to the monadic Ackermann class so as to preserve satisfiability and to increase the length of a formula of length n to $c^{n/\log n}$ at worst, for some constant c. By telescoping this reduction with the decision procedure of Section 10, a deterministic decision procedure of time complexity $c^{cn/\log n}$ is obtained, for some constant c.

The first steps simply reduce $\mathscr{F}$ to a subclass of $\mathscr{F}$. These steps result in only a linear increase of the length of a formula.

A. *Eliminate Initial Existentially Quantified Variables*

Let F be a formula $\exists z_1 \cdots \exists z_k \forall y_1 \exists x_1 \cdots \exists x_m \forall y_2 F_0$, where F_0 is quantifier-free. The goal of this step is to move the quantifiers $\exists z_1, \ldots, \exists z_k$ to between the universal quantifiers, where the rest of the existential quantifiers are located.

(a) For each variable z_i $(1 \leqslant i \leqslant k)$ introduce a dyadic predicate letter Z_i. (Intuitively, $Z_i t_1 t_2$ is to be true if and only if $t_1 = t_2 = z_i$.) Let F_1 be the formula

$$Z_1 z_1 z_1 \wedge \cdots \wedge Z_k z_k z_k.$$

(b) For each (necessarily dyadic) predicate letter P and each i, $1 \leqslant i \leqslant k$, such that F cointains an atomic formula $Pz_i v$ for some variable v other than $z_1,\ldots, z_k$, let P_{i*} be a new dyadic predicate letter (intuitively, $P_{i*}t_1t_2$ is to be true if and only if $t_1 = t_2$ and Pz_it_1 is true). Similarly, for each P and j such that F contains an atomic formula Pvz_j for some v not among $z_1,\ldots, z_k$, let P_{*j} be a new dyadic predicate letter; and for each P, i, and j such that Pz_iz_j is an atomic formula appearing in F, let P_{ij} be a new dyadic predicate letter. ($P_{*j}t_1t_2$ is to be true if and only if $t_1 = t_2$ and Pt_1z_j is true; and $P_{ij}t_1t_2$ is to be true if and only if $t_1 = t_2$ and Pz_iz_j is true.)

(c) Let F_0' be the result of replacing in F_0

(i) each occurrence of an atomic formula Pz_iv, where v is not among $z_1,\ldots, z_k$, by $P_{i*}vv$;

(ii) each occurrence of an atomic formula Pvz_j, where v is not among $z_1,\ldots, z_k$, by $P_{*j}vv$;

(iii) each occurrence of an atomic formula Pz_iz_j by $P_{ij}y_1y_1$.

Note that F_0' contains no occurrence of any of the variables $z_1,\ldots, z_k$.

(d) Let F_2 be the conjunction of the following three formulas:

(i) $\bigwedge_{P,j}(Z_iy_1y_1 \to (P_{i*}y_2y_2 \leftrightarrow Py_1y_2))$,

the conjunction over all the new predicate letters P_{i*};

(ii) $\bigwedge_{P,j}(Z_jy_2y_2 \to (P_{*j}y_1y_1 \leftrightarrow Py_1y_2))$,

the conjunction over all the new predicate letters P_{*j};

(iii) $\bigwedge_{P_{ij}}[(Z_iy_1y_1 \wedge Z_jy_2y_2 \to (P_{ij}y_1y_1 \leftrightarrow Py_1y_2)) \wedge (P_{ij}y_1y_1 \leftrightarrow P_{ij}y_2y_2)]$,

the conjunction over all the new predicate letters P_{ij}.

(e) Now let G be

$$\exists z_1 \cdots \exists z_k F_1 \wedge \forall y_1 \exists x_1 \cdots \exists x_m \forall y_2 (F_0' \wedge F_2).$$

Then G is satisfiable if and only if F is satisfiable; indeed any model of G is a model of F, and any model of F can be extended to a model of G by interpreting the new predicate letters as described. Moreover the length of G is linear in the length of F. Finally, G has a prenex form in the class $\mathscr{F}$ with all the existential quantifiers occurring between the two universal quantifiers.

B. *Eliminate Atomic Forms: First Step*

Now let $F = \forall y_1 \exists x_1 \cdots \exists x_m \forall y_2 F_0$ be any formula in $\mathscr{F}$ without initial existentially quantified variables. Then the atomic formulas of F are of the following forms, for various dyadic predicate letters P:

$$\begin{array}{lll} Py_1y_1 & Px_iy_1 & Py_2y_2 \\ Py_1x_j & Px_ix_j & \\ Py_1y_2 & Px_iy_2 & \end{array}$$

(but not Py_2y_1 or Py_2x_j). We now eliminate the forms Px_ix_j $(i \neq j)$ and Px_iy_1, leaving only Py_1y_1, Py_2y_2, Px_ix_i, Py_1y_2, Py_1x_j, and Px_iy_2. For each P and i such that Px_ix_j $(i \neq j)$ or Px_iy_2 occurs in F_0, introduce a new dyadic predicate letter P_i. Let F_0' be the result of replacing Px_ix_j $(i \neq j)$ by $P_iy_1x_j$ and Px_iy_1 by $P_iy_1y_1$ in F_0. Let

$$F_1 = \forall y_1 \exists x_1 \cdots \exists x_m \forall y_2 \left(F_0' \wedge \bigwedge_{P_i} (P_i y_1 y_2 \leftrightarrow Px_i y_2) \right)$$

the conjunction over all the new predicate letters P_i. Then F_1 is satisfiable if and only if F is satisfiable, and its length is linear in that of F.

C. *Eliminate Atomic Forms: Second Step*

Now let $F = \forall y_1 \exists x_1 \cdots \exists x_m \forall y_2 F_0$ be any formula in $\mathscr{F}$ without initial existentially quantified variables and having atomic formulas of only the following forms:

$$Py_1y_1 \qquad Px_ix_i \qquad Py_2y_2$$
$$Py_1x_j \qquad Px_iy_2$$
$$Py_1y_2$$

We now eliminate the form Py_1x_j. For each P and j such that this atomic formula appears in F, introduce a new dyadic predicate letter P_j. Let F_0' be the result of replacing Py_1x_j by $P_jx_jx_j$ in F_0, and let F_1 be

$$\forall y_1 \exists x_1 \cdots \exists x_m \forall y_2 \left(F_0' \wedge \bigwedge_{P_j} (P_j x_j y_2 \leftrightarrow Py_1 y_2) \right)$$

the conjunction over all the new predicate letters P_j. Again F_1 is satisfiable if and only if F is satisfiable and its length is linear in that of F.

D. *Eliminate Atomic Forms: Third Step*

Now let $F = \forall y_1 \exists x_1 \cdots \exists x_m \forall y_2 F_0$ be any formula in $\mathscr{F}$ without initial existentially quantified variables and having atomic formulas of only the following forms:

$$Py_1y_1 \qquad Px_ix_i \qquad Py_2y_2$$
$$Py_1y_2 \qquad Px_iy_2$$

We now elminate the form Px_ix_i. For each P such that for some i, this atomic formula appears in F, introduce a new dyadic predicate letter P'. Let F_0' be the result of replacing Px_ix_i by $P'x_iy_2$ in F_0, and let F_1 be

$$\forall y_1 \exists x_1 \cdots \exists x_m \forall y_2 \left(F_0' \wedge \bigwedge_{P'} (P' y_1 y_2 \leftrightarrow Py_1 y_1) \right)$$

the conjunction being over all the new predicate letters P'. Once again, F_1 is satisfiable if and only if F is satisfiable and its length is linear in that of F.

E. *Reduce to the Ackermann Class*

Now let $F = \forall y_1 \exists x_1 \cdots \exists x_m \forall y_2 F_0$ be any formula in $\mathcal{F}$ without initial existentially quantified variables and having atomic formulas of only the following forms:

$$Py_1 y_1 \qquad Px_i y_2$$
$$Py_1 y_2 \qquad Py_2 y_2.$$

We may assume without loss of generality that for every dyadic predicate letter P appearing in F, the atomic formulas $Py_1 y_1$, $Py_1 y_2$, and $Py_2 y_2$ actually appear in F. If not, we can add formulas such as $(\neg Py_1 y_1 \vee Py_1 y_1)$ to F_0 as conjuncts, and the length of F will increase only linearly.

Let $f_1,...,f_m$ be the monadic function signs associated with $x_1,...,x_m$ in the functional form of F. Thus each term in the Herbrand universe is of the form $f_{i_1}(f_{i_2}(\cdots (f_{i_n}(a)) \cdots))$, where a is some fixed constant sign. Because of the natural isomorphism between the Herbrand universe and the set of all strings over $\{f_1,...,f_m\}$, we denote terms by strings: If w is the string $f_{i_1} \cdots f_{i_n}$, then $\mathbf{w}$ is the term $f_{i_1}(\cdots (f_{i_n}(a)) \cdots)$.

We now construct a formula G in the Ackermann class which is satisfiable if and only if F is satisfiable. Actually, we construct not G but its functional form, which contains the same monadic function signs as does the functional form of F. F and G therefore have the same Herbrand universe.

Let F_1 be a full disjunctive normal form of F_0, and let F' be a formula obtained by prefixing to F_1 the quantifier prefix of F. Clearly F' is equivalent to F. For each disjunct D of F_1, let M_D be a new monadic predicate letter. (Intuitively, $M_D t_1$ is to be true if there is some term t_2 such that disjunct D of F_1 is true when y_1 has the value t_1 and y_2 has the value t_2.) The functional form of G is the conjunction of the four formulas G_1, G_2, G_3, G_4.

G_1 is

$$\bigvee_D M_D y$$

the disjunction being over all D such that

$$D \mid y_1 \sim (D \mid y_1, y_2)[y_2/y_1].$$

G_2 is

$$\bigwedge_{D_1, D_2} (\neg M_{D_1} y \vee \neg M_{D_2} y)$$

the conjunction being over all D_1, D_2 such that

$$D_1 \mid y_1 \nsim D_2 \mid y_1.$$

G_3 is

$$\bigwedge_{D_1, x_i} \left(M_{D_1} f_i(y) \to \bigvee_{D_2} M_{D_2} y \right)$$

the conjunction being over all D_1 and x_i, and the disjunction being over all D_2 such that

$$(D_2 \mid x_i, y_2)[x_i/y_1] \sim D_1 \mid y_1, y_2.$$

G_4 is

$$\bigwedge_{D_1, x_i} \left(M_{D_1} y \to \bigvee_{D_2} M_{D_2} f_i(y) \right)$$

the conjunction being over all D_1 and x_i, and the disjunction being over all D_2 such that

$$D_2 \mid y_1, y_2 \sim (D_1 \mid x_i, y_2)[x_i/y_1].$$

(An empty disjunction is to be construed as the constant "false.")

If n is the length of F, then the length of F_1 is at most $c^{n/\log n}$ for some c, and the length of G is polynomial in the length of F_1. Therefore to complete the proof it suffices to show that G is satisfiable if and only if F' is satisfiable. In the subsequent proof we write D^* for the formula that replaces disjunct D of F_1 in the functional form of F'.

If F' is satisfiable then G is satisfiable. For let $\mathscr{A}$ be any truth-assignment verifying each instance of the functional form of F'. Then define a truth-assignment $\mathscr{B}$ to be Herbrand expansion of G thus: For any predicate letter M_D and any term $\mathbf{w}$, $\mathscr{B}$ verifies $M_D\mathbf{w}$ if and only if for some term $\mathbf{u}$, $\mathscr{A}$ verifies $D^*[y_1/\mathbf{w}, y_2/\mathbf{u}]$. Then $\mathscr{B}$ verifies each instance of G_1, since for every $\mathbf{w}$ there is a D such that $\mathscr{A}$ verifies $D^*[y_1/\mathbf{w}, y_2/\mathbf{w}]$, and for this D it must be the case that

$$D \mid y_1 \sim (D \mid y_1, y_2)[y_2/y_1].$$

$\mathscr{B}$ verifies each instance of G_2, since if $\mathscr{A}$ verifies both $D_1^*[y_1/\mathbf{w}, y_2/\mathbf{u}_1]$ and $D_2^*[y_1/\mathbf{w}, y_2/\mathbf{u}_2]$ then necessarily $D_1 \mid y_1 \sim D_2 \mid y_1$.

$\mathscr{B}$ verifies each instance of G_3. A typical conjunct would have the form

$$M_{D_1} f_i(\mathbf{w}) \to \bigvee_{D_2} M_{D_2} \mathbf{w}$$

the disjunction being over all D_2 such that

$$(D_2 \mid x_i, y_2)[x_i/y_1] \sim D_1 \mid y_1, y_2.$$

If $\mathscr{B}$ verifies the antecedent then there is a term $\mathbf{u}$ such that $\mathscr{A}$ verifies $D_1^*[y_1/f_i(\mathbf{w}), y_2/\mathbf{u}]$. Consider then the instance of the functional form of F' in which $\mathbf{w}$ is the substituent for y_1 and $\mathbf{u}$ is the substituent for y_2. Some disjunct $D_2[y_1/\mathbf{w}, y_2/\mathbf{u}]$ of this instance is verified by $\mathscr{A}$. But then D_2 must have the property just stated,

$$(D_2 \mid x_i, y_2)[x_i/y_1] \sim D_1 \mid y_1, y_2,$$

since any atomic formula $Px_i y_2$ in D_2 becomes in $D_2^*[y_1/\mathbf{w}, y_2/\mathbf{u}]$ identical to the atomic formula that $Py_1 y_2$ becomes in $D_1^*[y_1/f_i(\mathbf{w}), y_2/\mathbf{u}]$.

$\mathscr{B}$ also verifies each instance of G_4, for reasons entirely analogous to those just given for G_3.

If G is satisfiable then F' is satisfiable. For this the argument is more complex. Let $\mathscr{A}$ be a truth-assignment verifying the Herbrand expansion of G. We define a mapping ϕ from pairs of terms to disjuncts of F_1. The mapping ϕ is defined in three phases.

(1) For each term $\mathbf{w}$, $\phi(\mathbf{w}, \mathbf{w})$ is some disjunct D such that $D \mid y_1 \sim (D \mid y_1, y_2)[y_2/y_1]$ and $\mathscr{A}$ verifies $M_D \mathbf{w}$. Such a disjunct is guaranteed to exist by G_1.

(2) Suppose that $f_i w$ is a suffix of u (not necessarily proper, i.e., $f_i w$ may be equal to u) and $\phi(f_i(\mathbf{w}), \mathbf{u})$ has been defined but $\phi(\mathbf{w}, \mathbf{u})$ has not. Let $D_1 = \phi(f_i(\mathbf{w}), \mathbf{u})$. Then $\phi(\mathbf{w}, \mathbf{u})$ is some disjunct D_2 such that $(D_2 \mid x_i, y_2)[x_i/y_1] \sim D_1 \mid y_1, y_2$ and $\mathscr{A}$ verifies $M_{D_2} \mathbf{w}$. Such a disjunct is guaranteed to exist by G_3.

(3) Suppose that $f_i w$ is not a suffix of u and $\phi(f_i(\mathbf{w}), \mathbf{u})$ has not been defined, but $\phi(\mathbf{w}, \mathbf{u})$ has. Let $D_1 = \phi(\mathbf{w}, \mathbf{u})$. Then $\phi(f_i(\mathbf{w}), \mathbf{u})$ is some disjunct D_2 such that $D_2 \mid y_1, y_2 \sim (D_1 \mid x_i, y_2)[x_i/y_1]$ and $\mathscr{A}$ verifies $M_{D_2} f_i(\mathbf{w})$. Such a disjunct is guaranteed to exist by G_4.

Since every pair of strings has some common suffix (possibly empty), (1)–(3) define ϕ unambiguously for all argument pairs (the nonconstructive "some" being resolved by some lexical ordering of the disjuncts).

Before proceeding further, we establish four crucial facts:

(a) For any w, $\phi(\mathbf{w}, \mathbf{w}) \mid y_1 \sim (\phi(\mathbf{w}, \mathbf{w}) \mid y_1, y_2)[y_2/y_1]$.

Proof. Immediate from clause (1) of the definition of ϕ.

(b) For any w, u_1, and u_2,

$$\phi(\mathbf{w}, \mathbf{u}_1) \mid y_1 \sim \phi(\mathbf{w}, \mathbf{u}_2) \mid y_1.$$

Proof. By inspection of (1)–(3), $\mathscr{A}$ verifies $M_{\phi(\mathbf{w},\mathbf{u})} \mathbf{w}$ for all $\mathbf{w}$ and $\mathbf{u}$. In particular, if $\phi(\mathbf{w}, \mathbf{u}_1) = D_1$ and $\phi(\mathbf{w}, \mathbf{u}_2) = D_2$, then $\mathscr{A}$ verifies both $M_{D_1} \mathbf{w}$ and $M_{D_2} \mathbf{w}$. But then $D_1 \mid y_1 \sim D_2 \mid y_1$ because $\mathscr{A}$ verifies each instance of G_2.

(c) For any $\mathbf{w}_1$, $\mathbf{w}_2$, and $\mathbf{u}$,

$$\phi(\mathbf{w}_1, \mathbf{u}) \mid y_2 \sim \phi(\mathbf{w}_2, \mathbf{u}) \mid y_2.$$

Proof. By inspection of (2) and (3). Note that as $\mathbf{u}$ remains fixed but $\mathbf{w}$ changes, $\phi(\mathbf{w}, \mathbf{u}) \mid y_2$ remains fixed in both (2) and (3).

(d) For any $\mathbf{w}$, $\mathbf{u}$, and any variable x_i,

$$(\phi(\mathbf{w}, \mathbf{u}) \mid x_i, y_2)[x_i/y_1] \sim \phi(f_i(\mathbf{w}), \mathbf{u}) \mid y_1, y_2.$$

Proof. By either (2) or (3), depending on whether $\phi(f_i(\mathbf{w}), \mathbf{u})$ is defined before or after $\phi(\mathbf{w}, \mathbf{u})$ (that is to say, depending on whether $f_i w$ is a suffix of u or not).

Now define a truth-assignment $\mathscr{B}$ to the Herbrand expansion of F' thus: $\mathscr{B}$ verifies $P\mathbf{wu}$ if and only if Py_1y_2 is a conjunct of $\phi(\mathbf{w}, \mathbf{u})$. We claim that $\mathscr{B}$ verifies each instance of the functional form of F'; specifically, for any $\mathbf{w}$, $\mathbf{u}$, if $\phi(\mathbf{w}, \mathbf{u}) = D$, then $\mathscr{B}$ verifies $D^*[y_1/\mathbf{w}, y_2/\mathbf{u}]$. This could fail to be the case only if there were $\mathbf{w}_1$, $\mathbf{w}_2$, $\mathbf{u}_1$, $\mathbf{u}_2$, and P such that $D_2^*[y_1/\mathbf{w}_2, y_2/\mathbf{u}_2]$ contained the atomic formula $P\mathbf{w}_1\mathbf{u}_1$ with the opposite sign from that of Py_1y_2 in $\phi(\mathbf{w}_1, \mathbf{u}_1)$, where $D_2 = \phi(\mathbf{w}_2, \mathbf{u}_2)$. We argue by cases that this cannot occur, depending on what atomic formula A of D_2 becomes $P\mathbf{w}_1\mathbf{u}_1$ in $D_2^*[y_1/\mathbf{w}_2, y_2/\mathbf{w}_2]$.

Case 1. A is Py_1y_2. Then $\mathbf{w}_2 = \mathbf{w}_1$, $\mathbf{u}_2 = \mathbf{u}_1$, and so $\phi(\mathbf{w}_1, \mathbf{u}_1) = \phi(\mathbf{w}_2, \mathbf{u}_2)$; but then Py_1y_2 cannot occur with opposite signs in this disjunct.

Case 2. A is Py_1y_1. Then $\mathbf{w}_2 = \mathbf{w}_1 = \mathbf{u}_1$. By (b), $\phi(\mathbf{w}_2, \mathbf{u}_2) \mid y_1 \sim \phi(\mathbf{w}_2, \mathbf{w}_2) \mid y_1$. By (a), $\phi(\mathbf{w}_2, \mathbf{w}_2) \mid y_1 \sim (\phi(\mathbf{w}_2, \mathbf{w}_2) \mid y_1, y_2)[y_2/y_1]$. Since both Py_1y_1 and Py_1y_2 actually occur in each disjunct, it follows that the occurrence of A in $\phi(\mathbf{w}_2, \mathbf{u}_2)$ is cosigned with the occurrence of Py_1y_2 in $\phi(\mathbf{w}_2, \mathbf{w}_2)$. But $\phi(\mathbf{w}_2, \mathbf{w}_2)$ is $\phi(\mathbf{w}_1, \mathbf{u}_1)$, since $\mathbf{w}_2 = \mathbf{w}_1 = \mathbf{u}_1$.

Case 3. A is Py_2y_2. Then $\mathbf{u}_2 = \mathbf{w}_1 = \mathbf{u}_1$. By (c), $\phi(\mathbf{w}_2, \mathbf{u}_2) \mid y_2 \sim \phi(\mathbf{u}_2, \mathbf{u}_2) \mid y_2$. By (a), $\phi(\mathbf{u}_2, \mathbf{u}_2) \mid y_1 \sim (\phi(\mathbf{u}_2, \mathbf{u}_2) \mid y_1, y_2)[y_2/y_1]$, and since Py_1y_1, Py_1y_2, and Py_2y_2 actually occur in each disjunct, it follows that the occurrence of Py_2y_2 in $\phi(\mathbf{w}_2, \mathbf{u}_2)$ is cosigned with the occurrences of Py_1y_1, Py_1y_2, and Py_2y_2 in $\phi(\mathbf{u}_2, \mathbf{u}_2)$, which is $\phi(\mathbf{w}_1, \mathbf{u}_1)$.

Case 4. A is Px_iy_2. Then $\mathbf{w}_1 = f_i(\mathbf{w}_2)$ and $\mathbf{u}_1 = \mathbf{u}_2$. Then it follows immediately from (d) that the occurrence of Px_iy_2 in $\phi(\mathbf{w}_2, \mathbf{u}_2)$ is cosigned with the occurrence of Py_1y_2 in $\phi(f_i(\mathbf{w}_2), \mathbf{u}_2)$, which is $\phi(\mathbf{w}_1, \mathbf{u}_1)$.

This completes the proof of Theorem 12.1. ∎

ACKNOWLEDGMENTS

I am grateful to Warren Goldfarb for suggesting the decision procedures of Theorems 5.1, 10.1, and 12.1. In the earlier version ([19]) of this paper, the reduction in Theorem 7.1 was of length order n^2; David Plaisted improved this to n log n, and we then sharpened Plaisted's method to obtain the present bound of n. Martin Fürer pointed out that the decision procedure of Theorem 10.1 is valid only for monadic formulas and supplied the upper bound given in Section 10 for the unrestricted Ackermann class.

REFERENCES

1. S. O. AANDERAA AND H. R. LEWIS, Linear sampling and the $\forall\exists\forall$ case of the decision problem, *J. Symbolic Logic* **39** (1974), 519–548.
2. W. ACKERMANN, Über die Erfüllbarkeit gewisser Zählausdrücke, *Math. Ann.* **100** (1928), 638–649.
3. W. ACKERMANN, "Solvable Cases of the Decision Problem," North-Holland, Amsterdam, 1954.
4. L. BERMAN, Precise bounds for Presberger arithmetic and the reals with addition, *in* "Proceedings, 18th Annual Symposium on Foundations of Computer Science, 1977," pp. 95–99.
5. P. BERNAYS AND M. SCHÖNFINKEL, Zum Entscheidungsproblem der mathematischen Logik, *Math. Ann.* **99** (1928), 342–372.
6. A. K. CHANDRA AND L. J. STOCKMEYER, Alternation, *in* "Proceedings, 17th Annual Symposium on Foundations of Computer Science, 1976," pp. 98–108.
7. S. A. COOK, The complexity of theorem-proving procedures, *in* "Proceedings, Third SIGACT Symposium, 1971," pp. 151–178.
8. S. A. COOK, A hierarchy for nondeterministic time complexity, *J. Comput. Systems Sci.* **7** (1973), 343–353.
9. B. DREBEN, A. S. KAHR, AND H. WANG, Classification of AEA formulas by letter atoms, *Bull. Amer. Math. Soc.* **68** (1962), 528–532.
10. B. S. DREBEN AND W. D. GOLDFARB, "The Decision Problem: Solvable Classes of Quantificational Formulas," Addison–Wesley, Reading, Mass., 1979.
11. M. J. FISCHER AND M. O. RABIN, Super-exponential complexity of Presberger arithmetic, *in* "Complexity of Computation," Vol. VII, "SIAM–AMS Proceedings," Amer. Math. Soc., Providence, R. I., 1974.
12. K. GÖDEL, Ein Spezialfall des Entscheidungsproblems der theoretischen Logik, Ergebnisse eines mathematischen Kolloquiums 2, pp. 27–28, 1932.
13. W. GOLDFARB, On the Gödel class with identity, *J. Symbolic Logic*, in press.
14. J. HARTMANIS AND R. E. STEARNS, On the computational complexity of algorithms, *Trans. Amer. Math. Soc.* **117** (1965), 285–306.
15. N. D. JONES AND A. L. SELMAN, Turing machines and the spectra of first-order formulas, *J. Symbolic Logic* **39** (1974),. 139–150.
16. L. KALMÁR, Über die Erfüllbarkeit derjenigen Zählausdrücke, welche in der Normalform zwei benachbarte Allzeichen enthalten, *Math. Ann.* **108** (1933), 466–484.
17. R. E. LADNER, R. J. LIPTON, AND L. J. STOCKMEYER, Alternating pushdown automata, *in* "Proceedings, 19th Annual Syposium on Foundations of Computer Science, 1978," pp. 92–106.
18. H. R. LEWIS, Description of restricted automata by first-order formulae, *Math. Systems Theory* **9** (1975), 97–104.
19. H. R. LEWIS, Complexity of solvable cases of the decision probem for the predicate calculus, *in* "Proceedings, 19th Annual Symposium on the Foundations of Computer Science, 1978," pp. 35–47.
20. H. R. LEWIS, "Unsolvable Classes of Quantificational Formulas," Addison–Wesley, Reading, Mass., 1979.
21. H. R. LEWIS, Satisfiability problems for propositional calculi, *Math. Systems Theory* **13** (1979), 45–53.
22. H. R. LEWIS AND C. H. PAPADIMITRIOU, "Elements of the Theory of Computation," Prentice-Hall, Englewood Cliffs, N. J., 1981, in press.
23. L. LÖWENHEIM, Über Möglichkeiten im Relativkalkül, *Math. Ann.* **76** (1915), 447–470; English translation in [33], pp. 228–251.
24. A. R. MEYER, Weak monadic second-order theory of successor is not elementary-recursive, MAC Technical Memo No. 38, M.I.T., 1973.
25. D. OPPEN, A upper bound on the complexity of Presburger arithmetic, *J. Comput. Systems Sci.* **16** (1978), 323–332.
26. D. A. PLAISTED, Complete problems in the first-order predicate calculus, unpublished.
27. W. V. O. QUINE, "Methods of Logic," Revised ed., Holt, Rinehart & Winston, 1963.

28. C. RACKOFF, "The Complexity of Theories of the Monadic Predicate Calculus," Technical Report, IRIA, 1975.
29. K. SCHÜTTE, Untersuchungen zum Entscheidungsproblem der mathematischen Logik, *Math. Ann.* **109** (1934), 572–603.
30. J. I. SEIFERAS, M. J. FISCHER, AND A. R. MEYER, Refinements of the nondeterministic time and space hierarchies, *in* "Proceedings, 14th IEEE Symposium on Switching and Automata Theory, 1973," pp. 130–137.
31. L. J. STOCKMEYER, "The Complexity of Decision Procedures in Automata Theory and Logic," Ph.D. Thesis, M.I.T., 1974, MAC TR–133.
32. J. VAN HEIJENOORT, "From Frege to Gödel: A Source Book in Mathematical Logic, 1879–1931," Harvard Univ. Press, Cambridge, Mass., 1971.