Tarskian Set Constraints

David A. McAllester
AT&T Labs
600 Mountain Ave
Murray Hill N.J. 07974
dmac@research.att.com

Robert Givan and Carl Witty
MIT Artificial Intelligence Laboratory
545 Technology Square
Cambridge Mass. 02139
rlg@ai.mit.edu
cwitty@ai.mit.edu

Dexter Kozen
Cornell University
Upson Hall
Ithaca, NY 14853
kozen@cs.cornell.edu

Abstract

We investigate set constraints over set expressions with Tarskian functional and relational operations. Unlike the Herbrand constructor symbols used in recent set constraint formalisms, the meaning of a Tarskian function symbol is interpreted in an arbitrary first order structure. We show that satisfiability of Tarskian set constraints is decidable in nondeterministic doubly exponential time. We also give complexity results and open problems for various extensions of the language.

1. Introduction

There has been considerable interest recently in formalisms for describing and reasoning about sets. Here we consider a family of formalisms that have received surprisingly little attention. Consider a set expression of the form $f(C_1, \ldots, C_n)$ where $C_1, \ldots, C_n$ denote sets. In recent work on set constraints function symbols are interpreted as Herbrand constructors so that the set expression $f(C_1, \ldots C_n)$ denotes the set of terms $f(t_1, \ldots, t_n)$ where $t_1 \in C_1, \ldots, t_n \in C_n$. But an equally natural interpretation takes $f(C_1, \ldots, C_n)$ to be the set of values that can derived by applying the *meaning* of f to elements of the sets denoted by $C_1, \ldots, C_n$. For example, if $+$ denotes addition and O denotes the set of odd integers then we would expect $+(O, O)$ to denote all the integers that can be expressed as the sum of two odds, i.e., all the even integers. In general we can let the meaning of operations be determined by a first order structure in the standard way. We call set expressions under this form of semantics "Tarskian" to distinguish them from the "Herbrand" set expressions that have received considerable recent attention.

Tarskian set constraints seem fundamentally different from Herbrand set constraints. There does not seem to be any simple reduction of Tarskian set constraints to the monadic class. Since Tarskian set constraints are not restricted to Herbrand interpretations, induction principles for Herbrand interpretations do not apply. It turns out that Tarskian set constraints are closely related to modal logics. Before stating our main results on Tarskian constraints we review some work on set calculi. We organize the review around four classes of set calculi — Herbrand set constraints, modal logics, AI concept languages, and Tarskian set constraints.

Herbrand set constraints involve set expressions generated by the following grammar.

$$C ::= X \mid f(C_1, \ldots C_n) \mid C_1 \cup C_2 \mid \neg C$$

A set expression of the form $f(C_1, \ldots, C_n)$ is taken to denote the set of all terms $f(t_1, \ldots, t_n)$ with $t_i \in C_i$. A set constraint is an expression of the form $C_1 \subseteq C_2$. Herbrand set constraints are largely inspired by applications to the static analysis of computer programs [19, 18, 13, 3]. The problem of determining satisfiability of a finite set of Herbrand set constraints is known to be complete for nondeterministic exponential time [1, 4]. The problem remains decidable in nondeterministic exponential time if one adds both negative constraints, i.e., $C_1 \not\subseteq C_2$, [2, 7], and projection functions [8].

Modal logics involve formulas which are true or false of possible worlds in Kripke structures. Equivalently, each formula of a modal logic can be taken to denote the set of worlds in which it is true. Since modal formulas denote sets, modal logics can be viewed as set calculi. Propositional dynamic logic (PDL) [12, 29] and the modal μ-calculus [22] are particularly significant modal logics. If R is a binary relation symbol and C is a set expression then in both these logics the set expression $[R]C$ denotes the set

$\{x : \forall y \;\; R(x,y) \rightarrow y \in C\}$. The set expression $<R>\; C$ is defined analogously to denote $\{x : \exists y \; R(x,y) \wedge y \in C\}$. The modal μ-calculus allows for recursively defined set expressions of the form $\mu X.C[X]$ where X is a set variable and $C[X]$ is a set expression in which every occurrence of X in $C[X]$ occurs inside an even number of negation signs. PDL can be seen as a restriction of the modal μ-calculus which has much simpler decision procedures and yet is sufficiently expressive to cover many applications. Satisfiability for both PDL and the modal μ-calculus are known to be complete for deterministic exponential time [32, 10, 30].

AI concept languages have been developed for knowledge representation in expert systems [5, 31]. The set expressions of concept languages are constructed from set variables and relation variables using a variety of compositional mechanisms. For example, the expression $\forall R.C$ where R is a relation expression and C is a set expression denotes the set $\{x : \forall y \;\; R(x,y) \rightarrow y \in C\}$ (and hence is a syntactic variant of $[R]C$). For the most part these languages can be viewed as fragments of PDL [6, 15, 14]. However, many of these languages have satisfiability problems in P, NP, or PSPACE [9]. Also, concept languages often include cardinality primitives which appear not to be expressible in PDL. Furthermore, there is a natural relationship between certain concept languages and Montague grammar for natural language. In particular the set expression $R(\texttt{every}\ C)$ is taken to be the set $\{x : \forall y \in C \;\; R(x,y)\}$. This provides a natural meaning for English verb phrases such as "contains every prime number." One simple but expressive Montagovian concept language has a polynomial time satisfiability problem [27].

Tarskian set expressions have been studied by Jónnson and Tarski in the framework of Boolean algebra with operations [20, 21]. In the work of Jónnson and Tarski the operation f in the expression $f(C_1, \ldots, C_n)$ actually denotes a relation on $n+1$ arguments. More specifically, $f(C_1, \ldots, C_n)$ denotes $\{y : \exists x_1 \in C_1, \ldots, x_n \in C_n : \langle x_1, \ldots, x_n, y\rangle \in f\}$. One can think of f as a nondeterministic operation — for any given tuple of inputs there is a set of possible outputs. Jónnson and Tarski's main result is a variant of the Stone representation theorem which can be viewed as a completeness theorem for an algebraic axiomatization. They did not study decision theoretic complexity issues. Representation theorems for subclasses of Boolean algebras with operations have recently been studied in a general setting by Goldblatt [17]. Kozen [23] has recently obtained a Stone duality in the context of Herbrand set constraints between the algebra of set constraints and the topological term automata of [24, 25].

Here we consider a superset of the original set expressions studied by Jónnson and Tarski. We make a syntactic distinction between deterministic and nondeterministic operation symbols corresponding to classical function symbols and relation symbols respectively. We use this nonstandard terminology so that we can write set expressions of the form $f(C_1, \ldots, C_n)$ where f is an operation symbol (either deterministic or nondeterministic). We also allow least fixed point expressions. The complete grammar of our Tarskian set expressions is as follows.

$$C ::= X \mid f(C_1, \ldots, C_n) \mid C_1 \cup C_2 \mid \neg C \mid \mu X.C$$

In the above grammar f can be either deterministic or nondeterministic and may take no arguments, i.e., be a constant symbol. $\mu X.C$ is restricted so that X can only occur inside an even number of negation symbols in C. We consider finite sets of constraints of the form $C_1 \subseteq C_2$ or $C_1 \not\subseteq C_2$.

In spite of the apparent naturality of Tarskian set constraints, their computational properties have not been widely studied. It is shown in [28] that satisfiability of non-recursive Tarskian set constraints not involving Boolean operations is decidable in cubic time (assuming unit time hash table operations). It is shown in [16] that satisfiability of constraints on expressions involving meets, joins, and monotone applications in an arbitrary lattice is similarly decidable in cubic time. The results of this paper are summarized in the table below. We categorize Tarskian set constraint satisfiability problems by the presence or absence of recursion (μ-sets), the presence or absence of functions (deterministic operations of arity at least one), and the presence or absence of constants (deterministic operations of arity 0). In all cases we allow nondeterministic operations (of all arities) and both positive and negative set constraints.

	Rec	Fun	Const	Lower Bound	Upper Bound
1.	-	-	-	EXPTIME	EXPTIME
2.	-	-	+	EXPTIME	EXPTIME
3.	-	+	-	NEXPTIME	NEXPTIME
4.	-	+	+	NEXPTIME	2-NEXPTIME
5.	+	-	-	EXPTIME	EXPTIME
6.	+	-	+	EXPTIME	?
7.	+	+	-	NEXPTIME	?
8.	+	+	+	Undecidable	?

The results in the first two lines of the table are proved using techniques similar to those used for PDL [29]. The lower bound in line three is proved using techniques similar to those used in proving NEXPTIME hardness for the monadic class [26]. The upper bound in line three is proved by a filtration-like argument. The lower bounds in lines one and three are proved in a more complete version of this paper available from the authors. The proofs of the upper bounds in lines one through three are discussed briefly in section 3.

Standard techniques fail for the fourth line, the case of non-recursive constraints with arbitrary operations. We show in section 3 that satisfiability for non-recursive Tarskian set constraints is decidable in nondeterministic doubly exponential time. Our procedure involves a reduction to a natural class of Diophantine constraints which we call *prequadratic*. We show that satisfiability for prequadratic Diophantine constraints is decidable in nondeterministic exponential time. However, we conjecture that prequadratic Diophantine satisfiability is in NP. If so, then we get a nondeterministic single exponential procedure for non-recursive Tarskian constraints.

The fifth line in the table corresponds to recursive constraints with nondeterministic operations. It turns out that constraint set satisfiability in this calculus is linear time equivalent to set expression satisfiability in the modal μ-calculus. We show in section 4 that constraint set satisfiability for this class is polynomial time reducible to closed set expression satisfiability in a calculus we call the Herbrand μ-calculus. Closed set expression satisfiability for the Herbrand μ-calculus is known to be decidable in exponential time.

Decision procedures for the modal μ-calculus can be viewed as consisting of two phases. The first phase can be viewed as a reduction of set expression satisfiability in the the modal calculus to set expression satisfiability in the closed Herbrand calculus. The second phase is a decision procedure for the closed Herbrand calculus. The formal justification for the first phase is rather elaborate [32]. Here we give an alternative reduction from the modal μ-calculus to the closed Herbrand μ-calculus with a simplified correctness proof.

It seems likely that techniques used in decision procedures for the the modal μ-calculus can be used to construct decision procedures for lines six and seven, although this has not yet been done.

The undecidability of the eighth line is proved by a reduction of Hilbert's tenth problem. The reduction, given in section 5, uses only intersection and union constraints (no negation) and only a single level of μ-quantification.

2. Basic Concepts

We assume a countably infinite collection of set variables and for each arity (number of arguments) an infinite number of deterministic and an infinite number of nondeterministic operation symbols of that arity. We consider set expressions generated by the following grammar.[1]

$$C ::= X \mid f(C_1,\ \ldots,\ C_n) \mid C_1 \cup C_2 \mid \neg C \mid \mu X.C$$

We take a first order structure $\mathcal{M}$ to be a domain (set) D plus an interpretation, denoted $\mathcal{M}(f)$, of each operation symbol f such that if f has arity n then $\mathcal{M}(f) \subseteq D^{n+1}$ and such that if f is deterministic then for all $x_1, \ldots x_n$ in D there exists exactly one y such that $\langle x_1,\ \ldots,\ x_n,\ y\rangle \in \mathcal{M}(f)$. A set variable interpretation over a first order structure $\mathcal{M}$ is a mapping from set variables to subsets of the domain of $\mathcal{M}$. If ρ is a set variable interpretation then $\rho[X := S]$ is the interpretation identical to ρ except that it interprets the variable X as the set S. For any set expression C, first order structure $\mathcal{M}$ with domain D, and set variable interpretation ρ over $\mathcal{M}$ we take $\mathcal{M}[\![C]\!]\rho$ to be a subset of D defined by the following conditions.

$$\begin{array}{lcl}
\mathcal{M}[\![X]\!]\rho & = & \rho(X) \\
\mathcal{M}[\![f(C_1,\ldots,C_n)]\!]\rho & = & \{y : \exists x_1 \in \mathcal{M}[\![C_1]\!]\rho, \ldots, x_n \in \mathcal{M}[\![C_n]\!]\rho : \\
 & & \quad \langle x_1,\ \ldots,\ x_n,\ y\rangle \in \mathcal{M}(f)\} \\
\mathcal{M}[\![C_1 \cup C_2]\!]\rho & = & \mathcal{M}[\![C_1]\!]\rho \cup \mathcal{M}[\![C_2]\!]\rho \\
\mathcal{M}[\![\neg C]\!]\rho & = & D - \mathcal{M}[\![C]\!]\rho \\
\mathcal{M}[\![\mu X.C]\!]\rho & = & \text{the least } S \text{ such that } S = \mathcal{M}[\![C]\!]\rho[X := S]
\end{array}$$

A constraint is an expression of the form $C_1 \subseteq C_2$ or $C_1 \not\subseteq C_2$. We say that the pair $\langle \mathcal{M},\ \rho\rangle$ satisfies the constraint $C_1 \subseteq C_2$ if $\mathcal{M}[\![C_1]\!]\rho \subseteq \mathcal{M}[\![C_2]\!]\rho$. $\langle \mathcal{M},\ \rho\rangle$ satisfies $C_1 \not\subseteq C_2$ if $\mathcal{M}[\![C_1]\!]\rho \not\subseteq \mathcal{M}[\![C_2]\!]\rho$. We say $\langle \mathcal{M},\ \rho\rangle$ satisfies a set Σ of constraints if $\langle \mathcal{M},\ \rho\rangle$ satisfies every member of Σ. A constraint set Σ is satisfiable if it is satisfied by some $\langle \mathcal{M},\ \rho\rangle$. We are interested in determining satisfiability of finite sets of constraints.

3. Constraints without Recursion

We now show that satisfiability of a finite set of Tarskian set constraints not involving recursion (μ-expressions) is decidable in nondeterministic doubly exponential time. Let Σ be a finite set of constraints not involving recursion. We say that a set expression C *occurs in* Σ if either C occurs as a top level set expression in a constraint in Σ or as a subexpression of such an expression. By abuse of notation we write $C \in \Sigma$ to indicate that C occurs in Σ. A Σ-type is a set τ of set expressions occurring in Σ and negations of expressions occurring in Σ satisfying the following conditions:

- If $(C_1 \cup C_2) \in \Sigma$ then τ contains $(C_1 \cup C_2)$ if and only if τ contains at least one of C_1 and C_2.
- If $C \in \Sigma$ then τ contains $\neg C$ if and only if τ does not contain C.
- If Σ contains $C \subseteq W$, and τ contains C, then τ contains W.

If $\langle \mathcal{M},\ \rho\rangle$ satisfies Σ and x is an element of the domain of $\mathcal{M}$ then $\{C \in \Sigma :\ x \in \mathcal{M}[\![C]\!]\rho\}$ is a Σ-type — we say

[1] The (deterministic or nondeterministic) operation symbol f must have arity n in expressions of the form $f(C_1,\ \ldots,\ C_n)$ and all occurrences of X in C in the expression $\mu X C$ must occur inside an even number of negations.

that it is the Σ-type inhabited by x. But not all Σ-types need be inhabited under a given pair $\langle \mathcal{M}, \rho \rangle$. Each interpretation yields a particular set of inhabited types. The set of inhabited types must satisfy certain conditions. We say that a type τ is a *possible output* of operation symbol f applied to types σ_1, ..., σ_n, written $f(\sigma_1, \ldots, \sigma_n) \rightsquigarrow \tau$, provided that τ does not contain a set expression of the form $\neg f(C_1, \ldots, C_n)$ where $C_i \in \sigma_i$ for each C_i. A set S of Σ-types is called *locally consistent* if:

- For each negative constraint $U \not\subseteq W$ in Σ there is a type in S which contains U but not W.
- If a type τ in S contains $f(C_1, \ldots, C_n)$ then there exist types σ_1, ..., σ_n in S such that $f(\sigma_1, \ldots, \sigma_n) \rightsquigarrow \tau$ and $C_i \in \sigma_i$.
- For each deterministic operation symbol f and Σ-types σ_1, ..., σ_n in S, where n is the arity of f, there exists a Σ-type τ in S such that $f(\sigma_1, \ldots, \sigma_n) \rightsquigarrow \tau$.
- Each constant (deterministic operation of arity zero) is contained in exactly one type in S.

If Σ does not contain recursion and either does not contain constants (deterministic operations of arity 0) or does not contain functions (deterministic operations of arity at least one) then Σ is satisfiable if and only if there exists a locally consistent set of Σ-types. In the case where neither constants or functions are present one can start with all Σ-types and iteratively remove those violating the second condition. We then have that Σ is satisfiable if and only if the resulting set of Σ-types satisfies the first condition. This gives a deterministic exponential time decision procedure. If constants are present then we must nondeterministically guess a unique Σ-type for each constant. However, this involves only polynomially many nondeterministic choices and hence the space of all possible guesses can be searched in deterministic exponential time. So we again get a deterministic exponential time procedure. When functions are present (but not constants) we can again start with all Σ-types and remove types violating the second and third conditions. However when the third condition is violated we have a choice of removing any one of the types σ_1, ..., σ_n. This gives a nondeterministic exponential time procedure.

When both functions and constants are present we must consider additional cardinality constraints. Consider the constraints $c_1 \cup c_2 \subseteq f(c_3)$ and $c_1 \not\subseteq c_2$ where c_1, c_2, c_3 and f are all deterministic. This constraint set has a locally consistent set of types but it is not satisfiable because $f(c_3)$ must be a singleton set while $c_1 \cup c_2$ must contain two elements.

Consider an element x which inhabits a Σ-type σ under $\langle \mathcal{M}, \rho \rangle$. For each application expression $f(C_1, \ldots, C_n)$ in σ there must exist elements $y_1 \in \mathcal{M}[\![C_1]\!]\rho$, ..., $y_n \in \mathcal{M}[\![C_n]\!]\rho$ such that $\langle y_1, \ldots, y_n, x \rangle \in \mathcal{M}(f)$. The values y_1, ..., y_n can be viewed as "predecessors" of x which "witness" the fact that x is in the set $f(C_1, \ldots, C_n)$. The predecessors y_1, ..., y_n each have Σ-types τ_1, ..., τ_n. We now define a *range expression* (over Σ) to be an expression of the form $f(\sigma_1, \ldots, \sigma_n)$, where $\sigma_1, \ldots, \sigma_n$ are Σ-types and f is an operation symbol. We say that a domain element d of a model $\mathcal{M}$ *inhabits* a range expression $f(\sigma_1, \ldots, \sigma_n)$ if there are some domain elements $d_1, \ldots, d_n$ inhabiting Σ-types $\sigma_1, \ldots, \sigma_n$ respectively such that $\langle d_1, \ldots, d_n, d \rangle \in \mathcal{M}(f)$.

Simply writing and solving inequality constraints on the cardinalities of the sets of inhabitants of the range expressions is still not enough to force the existence of a model. Consider the constraints $c_4 \not\subseteq c_5$, $c_4 \cup c_5 \subseteq f(c_1 \cup c_2)$, and $c_4 \cup c_5 \subseteq f(c_1 \cup c_3)$. These constraints are satisfiable but in any model we will have that $f(c_2) = f(c_3)$. The constraint set becomes unsatisfiable if we add $c_4 \cup c_5 \subseteq f(c_2 \cup c_3)$. There exists a locally consistent set of Σ-types for all three constraints. In any such set of Σ-types there will exists a constant c such that $f(c)$ appears in both the (unique) type containing c_4 and the (distinct) type containing c_5. The types for c_4 and c_5 appear locally consistent even if cardinally constraints on Σ-types and range expressions are considered. Furthermore, each individual constraint of the form $U \subseteq V$ appears consistent with cardinalities. More sophisticated cardinality constraints are needed.

We define a *predecessor justification* over Σ for a set $f(C_1, \ldots, C_n)$ to be a range expression $f(\tau_1, \ldots, \tau_n)$ over Σ such that $C_i \in \tau_i$. A *Σ-predecessor-type* is a pair $\langle \sigma, \Delta \rangle$ where σ is a Σ-type and Δ is a mapping from function applications appearing in the type σ to range expressions over Σ such that for any function application U in σ we have that $\Delta(U)$ is a predecessor justification of U and $\Delta(U) \rightsquigarrow \sigma$. An object x inhabits a Σ-predecessor-type $\langle \sigma, \Delta \rangle$ in a model $\mathcal{M}$ if x inhabits σ and, for each application expression U in σ, x inhabits $\Delta(U)$.

Each domain element will inhabit at least one predecessor type. However, an application expression $f(C_1, \ldots, C_n)$ can often be justified in more than one way and it is possible for a given domain element to inhabit many different predecessor types. In order to construct simple Diophantine constraints we introduce a choice function so that each domain element can be assigned a unique predecessor type.

Now let S be a set of Σ-types. For each type $\sigma \in S$ let z_σ be a variable representing the number of inhabitants of σ. For each range expression $f(\tau_1, \ldots, \tau_n)$ with $\tau_i \in S$ let $z_{f(\tau_1, \ldots, \tau_n)}$ be a variable representing the number of domain members inhabiting $f(\tau_1, \ldots, \tau_n)$. For each Σ-predecessor-type $\langle \sigma, \Delta \rangle$ let $z_{\langle \sigma, \Delta \rangle}$ be a variable representing the number of individuals whose selected predecessor type is $\langle \sigma, \Delta \rangle$. We define $D(S)$ to be following system of

Diophantine constraints.

$$\begin{array}{lcl}
z_\sigma & \geq & 1 \\
z_\sigma & = & \sum_{\langle \sigma,\ \Delta\rangle} z_{\langle \sigma,\ \Delta\rangle} \\
z_{f(\tau_1,\ \ldots,\ \tau_n)} & \geq & \sum_{\langle \sigma,\ \Delta\rangle:\ \exists U \in \sigma:\ \Delta(U) = f(\tau_1,\ \ldots,\ \tau_n)} z_{\langle \sigma,\ \Delta\rangle} \\
z_{f(\tau_1,\ \ldots,\ \tau_n)} & \leq & \Pi_i z_{\tau_i} \text{ for } f \text{ deterministic}
\end{array}$$

Theorem: A set Σ of non-recursive Tarskian constraints is satisfiable if and only if there exists a locally consistent set S of Σ-types such that the constraint set $D(S)$ is satisfiable over the positive integers plus ∞.

Without loss of generality we can assume that all operations take at most two arguments (larger arity operations can be encoded using a pairing function). Under this assumption it is possible to show that the size of $D(S)$ is only singly exponential in the size of Σ. It suffices to show that the number of predecessor types $\langle \sigma,\ \Delta\rangle$ is only singly exponential. Since the number of Σ-types σ is singly exponential, it suffices to show that the the number of functions Δ from class expressions in Σ to range expressions is singly exponential. But assuming binary operation symbols, the number of such functions is bounded by by $(|\Sigma|2^{4|\Sigma|})^{|\Sigma|}$ which is $2^{O(|\Sigma|^2)}$ and hence singly exponential.

We can eliminate the consideration of ∞ in $D(S)$ by nondeterministically guessing which variables are infinite and folding this guess into the constraints. The result is a set of Diophantine constraints over positive integers. By repeatedly replacing constraints of the form $z \leq xyw$ by $z \leq xu$, $u \leq yw$, we can arrive at a system of linear constraints on nonnegative integers plus a set of constraints of the form $x \leq yz$. We will call such a systems of constraints *prequadratic*.

Prequadratic Decidability Theorem: The problem of determining the satisfiability of a prequadratic set of Diophantine inequalities is solvable in nondeterministic exponential time.

Proof: We give a nondeterministic procedure for accepting solvable prequadratic constraint systems. The linear inequalities in the given system can be converted into an equisatisfiable set of Diophantine equations $Ax = B$ by introducing new "slack" variables. Call a variable x_i *bounded* in $Ax = B$ if there exists a finite upper bound on the value of x_i over all *rational* solutions to $Ax = B$. We can use linear programming (over the rationals) to determine which variables are bounded. Our nondeterministic procedure can now guess the values of the bounded variables. We can then replace each bounded variable by the guessed value giving a simplified problem. In substituting in the guesses, some of the nonlinear constraints become linear and must be added to the resulting linear sub-problem, yielding residual linear and nonlinear subproblems in fewer variables. We repeat this process until either it reaches a solution (all variables have been assigned values), or the residual linear constraints become unsolvable (over the rationals), or a state is reached where the residual linear problem is solvable (over the rationals) and all residual variables are unbounded. In the latter case, if the residual linear problem $Ax = B$ is solvable over the nonnegative integers then we accept the original prequadratic problem as solvable. Otherwise we fail.

We now show that if the procedure accepts a prequadratic system of constraints then that constraint set is solvable. If the procedure accepts then there exists a residual linear problem $Ax = B$, solvable over nonnegative integers, and where each variable is unbounded over the rationals, plus a residual set of nonlinear constraints. Let β be a solution to $Ax = B$ over the nonnegative integers. It is a fact of linear programming that if all variables are unbounded over the rationals then there must be a nonnegative rational solution α to $Ax = 0$ such that all components of α are nonzero. We can assume without loss of generality that α is integral because any nonintegral α can be made integral by multiplying by an appropriate constant. The vector $\beta + c\alpha$ is a solution to $Ax = B$ for any c. For sufficiently large c this vector also satisfies all nonlinear constraints because in any constraint $x \leq yz$ we have that x grows linearly in c while zy grows quadratically in c.

Finally we show that this nondeterministic procedure terminates in exponential time. Consider a prequadratic set of m Diophantine inequalities over n variables where the largest constant has b bits. An analysis of the maximum possible upper bound that can be imposed by a system of linear constraints shows that the binary representation of the value of a bounded variable can contain at most $O(bn \log n)$ bits. After k guesses the largest constant in the residual linear problem has at most $O(b(cn \log n)^{k+1})$ bits for some constant c. Since the number of guesses is bounded by n, we get an exponential upper bound on the size of the numbers appearing in the sequence of linear problems examined by the procedure. Since all the linear programming operations over the rationals can be done in polynomial time, and since integer programming is in NP, we get a nondeterministic exponential running time.■

The combination of the two theorems above yields a nondeterministic doubly exponential time procedure. We conjecture that satisfiability of prequadratic Diophantine equations is in NP. If so, we get nondeterministic singly exponential time.

4. Constraints without Determinism

In this section we consider Tarskian set constraints with recursive set expressions but without deterministic operation symbols. Constraint set satisfiability in this calculus turns out to be linear time equivalent to set expression satisfiability in the modal μ-calculus. Here we give a linear time reduction from Tarskian constraint set satisfiability without determinism to set expression satisfiability in a calculus we call the Herbrand μ-calculus. The Herbrand μ-calculus is known to be decidable in exponential time.

We say that a Tarskian set expression C is *satisfiable* if there exists $\langle \mathcal{M},\ \rho\rangle$ such that $\mathcal{M}[\![C]\!]\rho$ is nonempty. For any set Σ of Tarskian set constraints we define define $C[\Sigma]$ to be the following set expression.

$$\mu X.\ \left[\ \begin{array}{c} (\neg U_1 \cap W_1) \cup \ldots \cup (\neg U_n \cap W_n) \cup \\ \bigcup_{i,j}\ \ f_i(T,\ \ldots,\ T,\ X,\ T,\ \ldots,\ T) \end{array}\ \right]$$

Here X is a set variable not occurring in Σ, $W_1 \subseteq U_1$, …, $W_n \subseteq U_n$ are the positive set constraints in Σ, T is the set expression $Z \cup \neg Z$ for some arbitrary Z, and $f_i(T,\ \ldots,\ T,\ X,\ T,\ \ldots,\ T)$ ranges over all set expressions where f_i is an operation appearing in Σ and X occurs at argument j. Intuitively, we have $x \in C[\Sigma]$ if there exists a y reachable by inverse operations from x such that y violates a positive constraint in Σ. If $x \in \neg C[\Sigma]$ then the positive constraints in Σ are satisfied at all points reachable by inverse operations from x. If $\langle \mathcal{M},\ \rho\rangle$ satisfies Σ then $\mathcal{M}[\![C[\Sigma]]\!]\rho$ is the empty set.

It is easy to determine whether Σ is satisfied by the empty model (the model with the empty domain). For the nonempty case we have the following lemma.

Lemma: If Σ is a set of Tarskian constraints not involving deterministic operations then Σ is satisfiable by a nonempty model if and only if the set expression

$$f(U_1 \cap \neg W_1 \cap \neg C[\Sigma],\ \ldots,\ U_i \cap \neg W_i \cap \neg C[\Sigma]) \cap \neg C[\Sigma]$$

is satisfiable where f is a fresh operation symbol and where $U_1 \not\subseteq W_1, \ldots, U_n \not\subseteq W_n$ are all the negative constraints in Σ.

Proof: First suppose $\langle \mathcal{M},\ \rho\rangle$ satisfies Σ. For each negative constraint select a y_i such that $y_i \in U_i \cap \neg W_i$. Now interpret f as the operation containing the single tuple $\langle y_1,\ \ldots,\ y_n,\ x\rangle$ where x is an arbitrary element of the model. Now x is the desired element of the above class expression. Conversely suppose that $x \in \mathcal{M}[\![C]\!]\rho$ where C is the above class expression. Let $\mathcal{F}$ be the set of operation symbols appearing in C. Define the *inverse closure* of x in $\mathcal{M}$ under $\mathcal{F}$ to be the least subset S of the domain of $\mathcal{M}$ such that $x \in S$ and if $z \in S$ and $\langle y_1,\ \ldots,\ y_n,\ z\rangle \in \mathcal{M}(f)$ for $f \in \mathcal{F}$ then $y_i \in S$ for each y_i. The *inverse closure substructure* of $\mathcal{M}$ generated by x and $\mathcal{F}$ is the model $\mathcal{M}'$ whose domain is the inverse closure of x in $\mathcal{M}$ under $\mathcal{F}$ and such that for each nondeterministic operation $f \in \mathcal{F}$ we have that $\mathcal{M}'(f)$ is the restriction of the relation $\mathcal{M}(f)$ to the domain of $\mathcal{M}'$. We can now show by structural induction on a class expression C involving only operations in $\mathcal{F}$ that for any y in the domain of $\mathcal{M}'$, and set variable interpretation ρ over $\mathcal{M}$, we have $y \in \mathcal{M}[\![C]\!]\rho$ if and only if $y \in \mathcal{M}'[\![C]\!]\rho'$ where $\rho'(X)$ is the intersection of $\rho(X)$ with the domain of $\mathcal{M}'$. Intuitively we can think of C as a predicate on objects which only "looks at" objects in the inverse closure of its given argument. Given this fact it is possible to show that if $x \in \mathcal{M}[\![\neg C[\Sigma]]\!]\rho$ then the inverse image substructure of $\mathcal{M}$ generated by x and $\mathcal{F}$ satisfies all positive constraints in Σ. It is easy to see that $\mathcal{M}'$ satisfies all negative constraints in Σ and hence satisfies Σ.■

This lemma fails if we allow deterministic operations. For example consider the constraints $\mathbf{T} \not\subseteq \mathbf{F}$ and $f(\mathbf{T}) \subseteq \mathbf{F}$ where f is deterministic and $\mathbf{T}$ and $\mathbf{F}$ denote the universal and empty sets respectively. The set expression $\mathbf{T} \cap \neg\mathbf{F} \cap \neg C[\Sigma]$ is satisfiable but the constraint set is not.

Set satisfiability in both the modal μ-calculus and the Tarskian μ-calculus are polynomial time reducible to set satisfiability in a language we call the Herbrand μ-calculus. All of these calculi include set variables, Boolean operations on sets, and least fixed point expressions of the form $\mu X.C[X]$ where X occurs positively in $C[X]$. The modal μ-calculus has no application expressions but instead has set expressions of the form $<R>\ C$ where R is a binary relation symbol. The set expression $<R>\ C$ denotes the set $\{x:\ \exists y \in C:\ \ R(x,y)\}$. The Tarskian μ-calculus consists of the Tarskian set expressions defined here but without deterministic operations. The Herbrand μ-calculus has same syntax as the Tarskian μ-calculus but with only deterministic operations which are interpreted over the fixed universe of (possibly infinite) Herbrand terms. The set expression $f(C_1,\ \ldots,\ C_n)$ denotes the set of (possibly infinite) terms of the form $f(t_1,\ \ldots,\ t_n)$ with $t_i \in C_i$. In the Herbrand calculus we only consider the satisfiability problem for closed set expressions (ones not containing free set variables).

The closed Herbrand μ-calculus seems most natural for understanding the exponential time satisfiability algorithms for set expressions in these calculi [32], [10], [30]. The Herbrand calculus is based on the Herbrand universe of possibly infinite terms over a given set of function symbols. This would seem to indicate a relationship between the Herbrand calculus and Herbrand set constraints. However, in traditional Herbrand set constraint problems we are concerned with the existence of certain *sets* of Herbrand terms while here we are concerned with the existence of a single (possibly infinite) term satisfying given constraints.

There are many interesting examples of term sets definable in the Herbrand μ-calculus. The expression $\mu X.a \cup f(X)$ is the set of all finite terms which are some number of applications of f to a. We let $\nu X.C[X]$, a greatest fixed point expression, be an abbreviation for $\neg\mu X.\neg C[\neg X]$. The expression $\nu X.f(X)$ denotes a singleton set containing the infinite term $f(f(f(\ldots)))$. We will abbreviate this expression as f^ω. Another interesting example is $\mu X.g^\omega \cup f(X) \cup g(X)$. This is the set of infinite terms constructed from monadic function symbols f and g that have only finitely many occurrences of f. One can similarly define the set of infinite terms constructed from f and g that have only finitely many occurrences of g. Any satisfiability testing procedure must be capable of determining that the intersection of these two term sets is empty. It is known that the Herbrand μ-calculus defines exactly those term sets definable by Rabin tree automaton, or alternatively by formulas of SnS (the second order theory of n successors) [11].

It is known that the modal μ-calculus can be reduced in linear time to the Herbrand μ-calculus. Here we factor this reduction through the Tarskian calculus. There is a trivial satisfiability preserving reduction from the modal μ-calculus to the Tarskian μ-calculus where $<R>\ C$ is translated to $R(C)$. The reduction from the Tarskian calculus to the Herbrand calculus is almost as simple syntactically but more difficult to prove correct. For any expression C of the Tarskian calculus we define $T(C)$ by the equations $T(Y) = Y$, $T(\neg C) = \neg T(C)$, $T((C_1 \cup C_2)) = (T(C_1) \cup T(C_2))$, $T(\mu X.C[X]) = \mu X.T(C[X])$, and

$$\begin{aligned}&T(f(C_1, \ldots, C_n))\\ &= \mu X\, f(T(C_1), \ldots, T(C_n)) \cup g(\mathbf{T}, X) \cup g(X, \mathbf{T})\end{aligned}$$

where X is a fresh set variable and g is a fresh function symbol.

We will show that if C is a closed Tarskian set expression then C is satisfiable if and only if $T(C)$ is satisfiable. Since free set variables can be replaced with set constants (nondeterministic operations of no arguments) it suffices to consider closed expressions. For an expression C of the Herbrand μ-calculus we define $[\![C]\!]\rho$ by analogy with $\mathcal{M}[\![C]\!]\rho$ — in the Herbrand calculus no model is required. If C is closed then we write $[\![C]\!]$ to denote the meaning of C independent of any variable interpretation.

First we show that if $T(C)$ is satisfiable then so is C. We say that subterm s of a (possibly infinite) term w is *g-accessible* from w if either s is w or w is of the form $g(u, v)$ where s is g-accessible from either u or v. Let $\mathcal{M}$ be the Tarskian model whose domain is the set of all (possibly infinite) Herbrand terms and such that $\mathcal{M}(f)$ is the set of tuples $\langle y_1, \ldots, y_n, x\rangle$ such that $f(y_1, \ldots, y_n)$ is g-accessible from x. We can show by induction on C that for any variable environment ρ mapping variables to sets of (possibly infinite) Herbrand terms we have $\mathcal{M}[\![C]\!]\rho$ equals $[\![T(C)]\!]\rho$. So if $[\![T(C)]\!]$ is nonempty then $\mathcal{M}[\![C]\!]$ is nonempty and hence C is satisfiable.

Now we prove the converse. This proof is essentially a simplification of the proof given in [32] that any satisfiable set expression of the model μ-calculus can be satisfied by a model with bounded branching. First we simplify the problem by converting every expression to a purely positive form. This is done by introducing conjunctions, greatest fixed points $\nu X.C$ and "disapplications" $[f](C_1, \ldots, C_n)$. We define $\mathcal{M}[\![\nu X.C]\!]\rho$ to be the greatest subset S of the domain of $\mathcal{M}$ such that $S = \mathcal{M}[\![C]\!]\rho[X := S]$. We define the meaning of disapplications by $[f](C_1, \ldots, C_n) = \neg f(\neg C_1, \ldots, \neg C_n)$. In the Tarskian calculus we have $x \in \mathcal{M}[\![[f](C_1, \ldots, C_n)]\!]\rho$ if and only if for every tuple $\langle y_1, \ldots, y_n\rangle$ such that $\langle y_1, \ldots, y_n, x\rangle \in \mathcal{M}(f)$ we have that $y_i \in \mathcal{M}[\![C_i]\!]\rho$ for at least one y_i. We can now eliminate negation from any closed expression using de Morgan's laws and the following rules to push negations down.

$$\neg\mu X.C[X] = \nu X.\neg C[\neg X]$$

$$\neg\nu X.C[X] = \mu X.\neg C[\neg X]$$

$$\neg f(C_1, \ldots, C_n) = [f](\neg C_1, \ldots, \neg C_n)$$

$$\neg[f](C_1, \ldots, C_n) = f(\neg C_1, \ldots, \neg C_n)$$

Since all recursion must be monotone, variables can not appear in negative contexts and negation disappears entirely.[2] For any set expression of either the Tarskian or Herbrand μ-calculus we let $pos[C]$ be the positive form of C achieved by pushing negations down using these rules. We can extend the translation of the Tarskian calculus to the Herbrand calculus to handle greatest fixed points and disapplications by $T(\nu X.C) = \nu X.T(C)$ and $T([f](C_1, \ldots, C_n)) = \nu X.[f](T(C_1), \ldots, T(C_n)) \cap [g](\mathbf{F}, X) \cap [g](X, \mathbf{F})$. We now have that $T(pos(C))$ is semantically equivalent to $T(C)$. So to prove that T preserves satisfiability we need only consider positive expressions.

To handle the transfinite nature of μ-calculi we introduce syntactically indexed fixed point expressions of the form $\mu_\beta X.C$ and $\nu_\beta X.C$ where β is any ordinal. The semantics of these expressions are defined by the following equations.[3]

$$\mathcal{M}[\![\mu_\beta X.C[X]]\!]\rho = \bigcup_{\alpha<\beta} \mathcal{M}[\![C[\mu_\alpha X.C[X]]]\!]\rho$$

$$\mathcal{M}[\![\nu_\beta X.C[X]]\!]\rho = \bigcap_{\alpha<\beta} \mathcal{M}[\![C[\mu_\alpha X.C[X]]]\!]\rho$$

One can show that $\mathcal{M}[\![\mu X.C]\!] = \mathcal{M}[\![\mu_\beta X.C]\!]$ where β is any ordinal larger than the cardinality of $\mathcal{M}$. The same

[2] If P is a zero-ary nondeterministic operation of the Tarskian calculus then we can think of $[P]$ as a syntactic variant of $\neg P$.

[3] In these equation β can be empty, in which case the empty union denotes the empty set and the empty intersection the entire domain of $\mathcal{M}$. β can also be either a limit or successor ordinal.

statement holds for greatest fixed point expressions. An unindexed fixed point expression $\mu X.C$ can be viewed as a syntactic variant of $\mu_\infty X.C$ where ∞ is the class of all ordinals. Intuitively, ∞ plays the role of a "largest ordinal". So we can assume that all fixed point expressions are indexed. An expression in which all fixed point expressions are indexed with ∞ (i.e., unindexed) will be called a *maximally indexed expression*.

We now define a *type* to be a set σ of positive closed expressions satisfying the following conditions.

- If $(C \cap W) \in \sigma$ then $C \in \sigma$ and $W \in \sigma$.
- If $C \cup W \in \sigma$ then either $C \in \sigma$ or $W \in \sigma$.
- If $\mu_\beta X.C[X] \in \sigma$ then $C[\mu_\alpha X.C[X]] \in \sigma$ for some $\alpha < \beta$.
- If $\nu_\beta X.C[X] \in \sigma$ then $C[\nu_\beta X.C[X]] \in \sigma$.

Finally, we define an *execution tree* to be a pair $\langle \sigma,\ \Delta \rangle$ such that σ is a type and Δ is a set of expressions of the form $f(\gamma_1,\ \ldots,\ \gamma_n)$ where each γ_i is (recursively) an execution tree. We will be interested in infinite execution trees. We write $C \in \gamma$ if γ is a tree of the form $\langle \sigma,\ \Delta \rangle$ with $C \in \sigma$. An execution tree is called *locally consistent* if for every subtree $\langle \sigma,\ \Delta \rangle$ we have that both σ and Δ are countable sets, for every $f(\gamma_1,\ \ldots,\ \gamma_n) \in \Delta$ and $[f](W_1,\ \ldots,\ W_n) \in \sigma$ there is some W_i such that $W_i \in \gamma_i$, and for every $f(W_1,\ \ldots,\ W_n) \in \sigma$ there is some $f(\gamma_1,\ \ldots,\ \gamma_n) \in \Delta$ such that for all W_i we have $W_i \in \gamma_i$.

Lemma: If C is a closed satisfiable Tarskian set expression then there exists a locally consistent execution tree γ such that $C \in \gamma$.

Proof: Suppose $x \in \mathcal{M}[\![C]\!]$. We say that a set of expressions Σ is true at x (in $\mathcal{M}$) if $x \in \mathcal{M}[\![W]\!]$ for all $W \in \Sigma$. For any countable set Σ of expressions true a point x we construct a locally consistent execution $E(\Sigma,\ x)$ whose root type contains Σ. Let σ be a countable type containing Σ and true at x. For each expression $f(C_1,\ \ldots,\ C_n)$ in σ construct an element of Δ as follows. Select points $\langle y_1,\ \ldots,\ y_n \rangle$ such that $\langle y_1,\ \ldots,\ y_n,\ x \rangle \in \mathcal{M}(f)$ and $y_i \in \mathcal{M}[\![C_i]\!]$. For each $[f](W_1,\ \ldots,\ W_n) \in \sigma$ select a W_i such that $y_i \in \mathcal{M}[\![W_i]\!]$. Let Σ_i consist of C_i and all selected W_i. Now add $f(E(\Sigma_1,\ y_1),\ \ldots,\ E(\Sigma_n,\ y_n))$ to Δ. Finally return the pair $\langle \sigma,\ \Delta \rangle$.■

We now map an execution tree γ to a term $t(\gamma)$ using the following conditions where a is a new constant.

$$t(\langle \sigma,\ \{\} \rangle) = a$$

$$t(\langle \sigma,\ \{f(\gamma_1,\ \ldots,\ \gamma_n)\} \cup \Delta_2 \rangle) = g(f(t(\gamma_1),\ \ldots,\ t(\gamma_n)),\ t(\langle \sigma,\ \Delta_2 \rangle))$$

The second rule is applied "fairly" so that if γ is $\langle \sigma,\ \Delta \rangle$ and $f(\gamma_1,\ \ldots,\ \gamma_n) \in \Delta$ then $f(t(\gamma_1),\ \ldots,\ t(\gamma_n))$ is g-accessible from $t(\gamma)$.

Lemma: There exists a well founded ordering $<$ on closed syntactic expressions such that

$$W < C \ \text{ for } W \text{ a closed proper subexpression of } C$$

$$C[\mu_\alpha X.C[X]] < \mu_\beta X.C[X] \ \text{ for } \alpha < \beta$$

$$C[\nu_\alpha X.C[X]] < \nu_\beta X.C[X] \ \text{ for } \alpha < \beta$$

Proof: We define the Fisher-Ladner closure of an expression C to be the least set $FL(C)$ of maximally indexed expressions such that $C' \in FL(C)$ where C' is the result of maximally indexing all fixed points in C, any closed subexpression of an element of $FL(C)$ is an element of $FL(C)$, if $\mu_\infty X.C[X] \in FL(C)$ then $C[\mu_\infty X.C[X]] \in FL(C)$ and if $\nu_\infty X.C[X] \in FL(C)$ then $C[\nu_\infty X.C[X]] \in FL(C)$. The set $FL(C)$ is finite — it has one member for each (possibly open) subexpression of C. We define the *rank* of an expression to be the level of nesting of recursion of *closed* subexpressions. We define the *signature* of an expression C to be the tuple $\langle \alpha_1,\ \ldots,\ \alpha_n \rangle$ where n is the largest rank of any expression in $FL(C)$ and α_i is the maximum index of all closed recursion subexpressions of C of rank i, or 0 if there is no such subexpression. The signature of $\mu_\beta X.C$ is of the form $\langle \alpha_1,\ \ldots,\ \alpha_{j-1},\ \beta,\ 0,\ \ldots,\ 0 \rangle$ where j is the rank of $\mu_\beta X.C$. The signature of an unrolling $C[\mu_\alpha X.C[X]]$ with $\alpha < \beta$ is $\langle \alpha_1,\ \ldots,\ \alpha_{j-1},\ \alpha,\ \gamma_1,\ \ldots,\ \gamma_k \rangle$. The second signature is lexicographically smaller than the first (given $\alpha < \beta$) and hence unrolling reduces signature. We order signatures first by length and then lexicographically within signatures of the same length. We order expressions lexicographically by signature then syntactic depth.■

Lemma: If γ is a locally consistent execution and $C \in \gamma$ then $t(\gamma) \in [\![T(C)]\!]$.

Proof: We define a ν-reindexing of an expression C to be any expression C' identical to C except for the indices of ν-expressions. We prove by induction on expressions using the ordering of the preceding lemma that if C is any ν-reindexing of an expression $C' \in \gamma$ then $t(\gamma) \in [\![T(C)]\!]$. To show the need for ν-reindexing we will explicitly give the proof for ν-expressions. Consider an expression $\nu_\beta X.C[X]$ which is a ν-reindexing of an expression $\nu_\delta X.C'[X] \in \gamma$. We have $C'[\nu_\delta X.C'[X]] \in \gamma$. Now consider any ordinal $\alpha < \beta$. By the induction hypothesis we have that $t(\gamma) \in [\![T(C[\nu_\alpha X.C[X])]\!]$. But we have that $[\![T(\nu_\beta X.C[X])]\!]$ is the intersection of all such sets so we have $t(\gamma) \in [\![\nu_\beta X.C[X]]\!]$. The other cases of the induction are straightforward given the above properties of the well founded ordering on expressions.■

Theorem: $T(C)$ is satisfiable if and only if C is satisfiable.

5 The Full Tarskian Calculus

In this section we show that satisfiability for full Tarskian set constraints (with recursion and arbitrary functions) is undecidable. The proof is by a reduction of Hilbert's tenth problem. The proof uses only set variables, constants, monadic functions, set unions and intersections (no complementation), and a single level of μ quantification.

> **Theorem:** Satisfiability for Tarskian set constraints with constants and monadic functions is undecidable.

Proof: Let Σ be a set of constraints of the form $n = 1$, $n = p+q$ or $n = pq$ where n, p and q range over nonnegative integers. It follows from the undecidability of Hilbert's tenth problem that satisfiability for such systems of constraints is undecidable. We reduce the Diophantine constraint set Σ to a set $T(\Sigma)$ of Tarskian set constraints as follows.

For each natural number variable n occurring in Σ we introduce a set variable X_n with the intention that the cardinality of X_n represents the value of n. For set expressions C and W we will use $C = W$ as an abbreviation for the two constraints $C \subseteq W$ and $W \subseteq C$. We will also use $|C| \leq |W|$ as an abbreviation for $C \subseteq f(W)$ where f is a fresh monadic function symbol. We will use $|C| = |W|$ as an abbreviation for $|C| \leq |W|$ and $|W| \leq |C|$. For any monadic function symbol s and class expression C we let $s^*(C)$ be an abbreviation for $\mu W.\ C \cup s(W)$, i.e., the set of things that can be gotten by applying s zero or more times to an element of C. For each variable n in Σ we introduce a constant symbol c_n and monadic function symbol s_n and add the constraints

$$X_n = s_n^*(c_n)$$

$$c_n \subseteq s_n(s_n^*(c_n)).$$

The first constraint states that X_n is the set containing c_n and all its transitive successors under s_n. The second constraint states that c_n is the successor of some element of $s_n^*(c_n)$ and therefore that the set X_n forms a loop under the successor function s_n. This implies that X_n is a finite set but does not otherwise constrain its cardinality. We now need to impose the constraints given in Σ.

If Σ contains the constraint $n = 1$ then $T(\Sigma)$ contains the constraint $X_n = c_n$. If Σ contains $n = p + q$ then we add the constraints

$$X_n = U \cup W$$
$$|X_p| = |U|$$
$$|X_q| = |W|$$
$$U \cap W \subseteq F$$

to $T(\Sigma)$ where C, U, and W are fresh set variables and F is the set expression $\mu X.\ X$. It remains only to express product constraints.

To handle the product case we use the notation $\forall x \in f^*(c)\ \ x = C[x]$ as an abbreviation for

$$f^*(c) = \mu X.(c \cap C[c]) \cup (f(X) \cap C[f(X)])$$

For example, $\forall x \in f^*(c)\ \ x = g(f(x))$ states that g is the inverse of f on the set $f^*(c)$. More generally, if there is only one occurrence of X in $C[X]$, and $C[X]$ is constructed purely from X and function symbols, then $\forall x \in f^*(c)\ \ x = C[x]$ has the obvious intended meaning. Suppose Σ contains $n = pq$. We add the following constraints to $T(\Sigma)$.

1. $X_n = f^*(c)$
2. $c \subseteq f(f^*(c))$
3. $X_p = g^*(c)$
4. $c \subseteq g(g^*(c))$
5. $X_q = h^*(c)$
6. $c \subseteq h(h^*(c))$
7. $f^*(c) = \mu X\ c \cup g(X) \cup h(X)$
8. $\forall x \in f^*(c)\ \ x = g'(g(x))$
9. $\forall x \in f^*(c)\ \ x = h'(h(x))$
10. $\forall x \in f^*(c)\ \ x = g'(h'(g(h(x))))$
11. $g^*(c) \cap h^*(c) = c$

Where c is a fresh constant and f, g, h, g' and h' are fresh monadic function symbols. Constraints 2, 4, and 6 imply that $f^*(c)$, $g^*(c)$, and $h^*(c)$ are all "loops". Constraint 7 implies that g and h are both functions mapping $f^*(c)$ into $f^*(c)$. Constraints 8, and 9 imply that g', and h' are inverses of g and h respectively on the set $f^*(c)$. Since both g and h are invertible they must both be bijections from $f^*(c)$ to itself. This implies that the inverses g' and h' are also bijections. Condition 10 implies that g and h commute on $f^*(c)$, i.e., $f(g(x))$ equals $g(f(x))$. Now consider $g^n(h^*(c))$. Since g is bijective, g^n is bijective. Note that $h(g^n(x))$ equals $g^n(h(x))$. So the mapping g^n is a bijection which "preserves h structure". Hence the set $g^n(h^*(c))$ is an h-loop with the same cardinality as $h^*(c)$. Since sets of the form $g^j(h^*(c))$ are h-loops they are either equal or disjoint. Suppose $g^j(h^*(c)) = g^k(h^*(c))$. Applying $(g')^j$ to both sides we get $h^*(c) = g^{k-j}(h^*(c))$. This implies that $g^{k-j}(c)$ must be in $h^*(c)$ and hence by condition 11 above we have $g^{k-j}(c) = c$. But this implies that k equals j mod $|g^*(c)|$. Hence for $k \neq j$ mod $|g^*(c)|$ we have $g^j(h^*(c))$ is disjoint from $g^k(h^*(c))$. Since all these sets are of size $|h^*(c)|$ we have $|f^*(c)| = |g^*(c)||h^*(c)|$. ∎

6. Conclusions

A wide variety of set calculi have been studied in the logic and computer science literature. Tarskian set expressions yield a natural set calculus which has received surprisingly little attention. We have answered a variety of questions concerning the computational complexity of Tarskian set constraints but several problems remain open. It seems likely that Tarskian set constraints without recursion (but with deterministic operations) can be solved in nondeterministic singly exponential time. This would follow from a demonstration that satisfiability of prequadratic Diophantine equations is in NP. The decidability of Tarskian set constraints with recursion and deterministic operations of arity at least 1, or with arity just zero, remains open. It seems likely that techniques used in decision procedures for the modal μ-calculus can be also be used to construct decision procedures for these cases, although this has not yet been done.

References

[1] A. Aiken, D. Kozen, M. Vardi, and E. Wimmers. The complexity of set constraints. In E. Börger, Y. Gurevich, and K. Meinke, editors, *Proc. 1993 Conf. Computer Science Logic (CSL'93)*, volume 832 of *Lect. Notes in Comput. Sci.*, pages 1–17. Eur. Assoc. Comput. Sci. Logic, Springer, September 1993.

[2] A. Aiken, D. Kozen, and E. Wimmers. Decidability of systems of set constraints with negative constraints. *Infor. and Comput.*, 1994. To appear. Also Cornell University Tech. Report 93-1362, June, 1993.

[3] A. Aiken, E. Wimmers, and T. Lakshman. Soft typing with conditional types. In *ACM Symposium on Principles of Programming Languages*, pages 163–173. Association for Computing Machinery, 1994.

[4] L. Bachmair, H. Ganzinger, and U. Waldmann. Set constraints are the monadic class. In *Proceedings, Eighth Annual IEEE Symposium on Logic in Computer Science*, pages 75–83, Montreal, Canada, 19–23 June 1993. IEEE Computer Society Press.

[5] R. Brachman and J. Schmolze. An overview of the kl-one knowledge representation system. *Computational Intelligence*, 9(2):171–216, 1985.

[6] D. Calvanese, M. Lenzerini, and D. Nardi. A unified framework for class-based representation formalisms. In *Proceedings of the Fourth International Conference on Principles of Knowledge Representation and Reasoning*, pages 109–119, May 1994.

[7] W. Charatonik and L. Pacholski. Negative set constraints with equality. In *Proc. 9th Symp. Logic in Computer Science*, pages 128–136. IEEE, July 1994.

[8] W. Charatonik and L. Pacholski. Set constraints with projections are in NEXPTIME. In *Proc. 35th Symp. Foundations of Computer Science*, pages 642–653. IEEE, November 1994.

[9] F. Donini, M. Lenzerini, D. Nardi, and W. Nutt. The complexity of concept languages. In *Proceedings of KR91*, pages 151–162. Morgan Kaufmann Publishers, 1991.

[10] E. Emerson and C. S. Jutla. The complexity of tree automata and modal logics of programs. In *Proc. 29th Ann. IEEE Symp. on Foundations of Computer Science*, pages 328–337. IEEE, 1988

[11] E. A. Emerson and C. S. Jutla. Tree automata, μ-calculus, and determinancy. In *Proc. 32nd Ann. IEEE Symp. on Foundations of Computer Science*, pages 368–377, 1991.

[12] M. Fischer and R. E. Ladner. Propositional dynamic logic of regular programs. *Journal of Comput. System Sci.*, 18(2):194–211, 1978.

[13] T. Frühwirth, E. Shapiro, M. Vardi, and E. Yardeni. Logic programs as types for logic programs. In *Proceedings, Sixth Annual IEEE Symposium on Logic in Computer Science*, pages 75–83. IEEE Computer Society Press, 1991.

[14] G. D. Giacomo and M. Lenzerini. Boosting the correspondence between description logics and dynamic logic. In *AAAI94*, pages 205–212, 1994.

[15] G. D. Giacomo and M. Lenzerini. Concept languages with number restrictions and fixed points and its relationship with mu-calculus. In *11th European Conf. on Artificial Intelligence*, pages 356–360, Aug. 1994.

[16] R. Givan and D. McAllester. New results on local inference relations. In *Principles of Knolwedge Representation and Reasoning: Proceedings of the Third International Conference*, pages 403–412. Morgan Kaufman Press, October 1992.

[17] R. Goldblatt. Varieties of complex algebras. *Annals of Pure and Applied Logic*, 44:173–242, 1989.

[18] N. Heintze and J. Jaffar. A decision procedure for a class of set constraints. In *Proceedings, Fifth Annual IEEE Symposium on Logic in Computer Science*, pages 42–51. IEEE Computer Society Press, 1990.

[19] N. Heintze and J. Jaffar. A finite presentation theorem for approximating logic programs. In *ACM Symposium on Principles of Programming Languages*, pages 197–209. Association for Computing Machinery, 1990.

[20] B. Jónnson and A. Tarski. Boolean algebras with operators. part I. *Amer. J. Math*, 73:891–939, 1951.

[21] B. Jónsson and A. Tarski. Boolean algebras with operators. part II. *Amer. J. Math*, 74:127–167, 1952.

[22] D. Kozen. Results on the propositional μ-calculus. *Theoretical Computer Science*, 27:333–354, 1983.

[23] D. Kozen. Logical aspects of set constraints. In *Proc. 1993 Conf. Computer Science Logic (CSL93)*. Eur. Assoc. Comput. Sci. Logic, September 1993.

[24] D. Kozen, J. Palsberg, and M. I. Schwartzbach. Efficient recursive subtyping. In *Proc. 20th Symp. Princip. Programming Lang.*, pages 419–428. ACM, January 1993.

[25] D. Kozen, J. Palsberg, and M. I. Schwartzbach. Efficient inference of partial types. *J. Comput. Syst. Sci.*, 49(2):306–324, October 1994.

[26] H. R. Lewis. Complexity results for classes of quantificational formulas. *J. of Comp. Syst. and Syst. Sci.*, 21:317–353, 1980.

[27] D. McAllester and R. Givan. Natural language syntax and first order inference. *Artificial Intelligence*, 56:1–20, 1992.

[28] D. McAllester and R. Givan. Taxonomic syntax for first order inference. *JACM*, 40(2):246–283, April 1993.

[29] V. R. Pratt. A near-optimal method for reasoning about actions. *J. Comput. System Sci.*, 20(2):231–254, 1980.

[30] S. Safra. On the complexity of ω-automaton. In *Proc. 29th Ann. IEEE Symp. on Foundations of Computer Science*, pages 319–327. IEEE, 1988.

[31] M. Schmidt-Schaub and G. Smalka. Attributive concept descriptions with complements. *Artificial Intelligence*, 47:1–26, 1991.

[32] R. S. Streett and E. A. Emerson. An automaton theoretic decision procedure for the propositional μ-calculus. *Information and Computation*, 81:249–264, 1989.