Tarskian Set Constraints are in NEXPTIME*

Pawel Mielniczuk and Leszek Pacholski

Institute of Computer Science
University of Wroclaw
{mielni,pacholsk}@tcs.uni.wroc.pl

Abstract. In this paper we show that satisfiability of Tarskian set constraints (without recursion) can be decided in exponential time. This closes the gap left open by D.A. McAllester, R. Givan, C. Witty and D. Kozen in [14].

Introduction

Set constraints have a form of inclusions between set expressions built over a set of set-valued variables, constants and function symbols. They have been used in program analysis and type inference algorithms for functional, imperative and logic programming languages [3], [11], [12], [15], [16], [18].
The systems of set constraints used for program analysis were considered as inclusion constraints over the Herbrand universe i.e. a solution consisted of a collection of subsets of the Herbrand universe. To distinguish them from set constraints studied here we shall call them the *Herbrand* set constraints.
The satisfiability problem for Herbrand set constraints have been studied by many authors including N. Heintze and J. Jaffar [10], A. Aiken and E.L. Wimmers [4], R. Gilleron, S. Tison, and M. Tommasi [7], L. Bachmair, H. Ganzinger, and U. Waldmann [5], A. Aiken, D. Kozen, M. Vardi, and E.L. Wimmers [1]. The strongest decidability results were obtained by A. Aiken, D. Kozen, and E. Wimmers [2], R. Gilleron, S. Tison and M. Tommasi [8], K. Stefansson [17], and by W. Charatonik and L. Pacholski [6].
Recently D.A. McAllester, R. Givan, C. Witty and D. Kozen [14] liberalized the notions of set constraints to so-called Tarskian set constraints over arbitrary first-order domain, with a link to modal-logics. Early work on Tarskian set constraints by R. Givan and D. McAllester stemmed from the work on artificial intelligence [9, 13].
D. McAllester, R. Givan, C. Witty and D. Kozen [14] gave a complexity analysis of systems of Tarskian set constraints depending on various parameters. They proved that the satisfiability problem for set constraints with *deterministic* operators of arbitrary arity (functions and constants) extended by recursion (μ-operator) is undecidable. They proved that for pure set constraints with deterministic constants but without deterministic function symbols of arity > 0, the satisfiability problem is EXPTIME-complete, and for pure set constraints with

* This research was partially supported by a KBN grant 8 T11C 029 13

deterministic function symbols of arity > 0, but without deterministic constants the satisfiability problem is NEXPTIME-complete. Finally they proved that for full (with deterministic constant and function symbols) set constraints without recursion the satisfiability problem is in 2-NEXPTIME and is NEXPTIME-hard, leaving the gap between the upper and the lower bound.
The approach to the upper bound was based on a reduction of the satisfiability problem to a problem of solving systems of so called prequadratic Diophantine inequalities. The result of reduction was of exponential size and the algorithm solving prequadratic Diophantine inequalities presented in [14] worked in non-deterministic exponential time, so the resulting set constraint algorithm worked in 2-NEXPTIME. D.A. McAllester, R. Givan, C. Witty and D. Kozen expressed a hypothesis that the satisfiability problem for systems of prequadratic Diophantine inequalities was in NP, which would give an algorithm of complexity matching the lower bound.
Here we give a NEXPTIME algorithm solving full systems of Tarskian set constraints (without recursion). We do it without a reduction to systems of prequadratic Diophantine inequalities.

1 Basic Definitions

We assume an infinite set of (deterministic and non-deterministic) operator symbols of each arity.

Definition 1. *1. A* set expression *is defined by the grammar:*

$$E ::= F(E_1, \ldots, E_n) \mid E_1 \cup E_2 \mid E_1 \cap E_2 \mid \neg E,$$

where F is an operator symbol of arity n.
2. By $\mathcal{M}$ we denote a first order structure with the universe M. Set expressions are interpreted in $\mathcal{M}$ as follows: $F^{\mathcal{M}}(E_1 \cup E_2) = F^{\mathcal{M}} E_1 \cup F^{\mathcal{M}} E_2$, $F^{\mathcal{M}}(E_1 \cap E_2) = F^{\mathcal{M}} E_1 \cap F^{\mathcal{M}} E_2$, $F^{\mathcal{M}}(\neg E_1) = M \setminus F^{\mathcal{M}} E_1$ and

$$F^{\mathcal{M}}(E_1, \ldots, E_n) = \{x \mid F^{\mathcal{M}}(x_1, \ldots, x_n) = x,\, x_i \in E_i^{\mathcal{M}},\, 1 \leq i \leq n\}.$$

If F is deterministic then, for all $x_1, \ldots, x_n \in \mathcal{M}$, there is exactly one $x \in \mathcal{M}$ such that $F^{\mathcal{M}}(x_1, \ldots, x_n) = x$.

A non-deterministic operator of arity zero can, from the point of view of satisfiability, be considered as a *set variable*. In the following we shall identify non-deterministic operators of arity zero with set variables. For this reason we did not explicitly mention variables in Definition 1. A deterministic operator is called a *function*, if it is of arity zero we call it a *constant*.

Definition 2. *A* constraint *is an expression of the form $E_1 \subseteq E_2$ (*positive constraint*) or $E_1 \not\subseteq E_2$ (*negative constraint*), where E_1, E_2 are set expressions. A* constraint *of the form $E_1 \subseteq E_2$ ($E_1 \not\subseteq E_2$) is satisfied in $\mathcal{M}$ if $E_1^{\mathcal{M}} \subseteq E_2^{\mathcal{M}}$ ($E_1^{\mathcal{M}} \not\subseteq E_2^{\mathcal{M}}$ respectively). We say that a set S of constraints is satisfiable, if there is a structure $\mathcal{M}$ such that every element of S is satisfied in $\mathcal{M}$.*

We consider the problem to determine whether a finite set of constraints is satisfiable.

2 Set Constraints as Flat Expressions

We reduce systems of set constraints to one inclusion constraint defined using a flat set expression, where *flat set expression* is a set expression involving terms of depth at most one.

Definition 3. *A* flat set expression *is defined by the following grammar:*

$$E ::= F(X_1, \ldots, X_n) \mid E_1 \cup E_2 \mid E_1 \cap E_2 \mid \neg E,$$

where $X_1, \ldots, X_n$ are set variables.

The *size* of a system S of set constraints is the number of symbols in S.

Proposition 1. *For every system S of set constraints of size n there is a flat set expression $\mathcal{E}$ of size polynomial in n such that S is equivalent to inclusion constraint $\mathcal{E} \subseteq \emptyset$. Moreover, we can assume that negation is applied only to expressions of the form $F(X_1, \ldots, X_n)$.*

Proof. Set expressions of depth greater than one can be eliminated from S as follows. If $F(E_1, \ldots, E_n)$ appears in S and E_i is not a variable, for some $0 < i \leq n$, then S is replaced by $S[X_i/E_i] \cup \{X_i \subseteq E_i,\, E_i \subseteq X_i\}$, where X_i is a new set variable and $S[X_i/E_i]$ is the result of replacing E_i by X_i everywhere in S.
A negative constraint $E_1 \not\subseteq E_2$ can be replaced by $c \subseteq E_1 \cap \neg E_2$, for a new constant symbol c. Finally, $E_1 \subseteq E_2$ is equivalent to $E_1 \cap \neg E_2 \subseteq \emptyset$, and $E_1 \subseteq \emptyset$, $E_2 \subseteq \emptyset$ is equivalent to $E_1 \cup E_2 \subseteq \emptyset$.
The last sentence follows by de Morgan rules. □

As an exercise the reader can check that a flat version of $f(a) \subseteq a$ is $(f(X) \cap \neg a) \cup (X \cap \neg a) \cup (\neg X \cap a) \subseteq \emptyset$.

3 Description of Structures

Let $\mathcal{E}$ be a flat set expression. Denote by Σ_0 the set of variables occurring in $\mathcal{E}$, and by Σ_1 - the set of set expressions of the form $F(X_1, \ldots, X_n)$, where F occurs in $\mathcal{E}$ and $X_1, \ldots, X_n \in \Sigma_0$. Define $\Sigma = \Sigma_0 \cup \Sigma_1$.
A Σ-type (Σ_0-type, Σ_1-type) is a set expression τ of the form $\bigcap T$ where T is a set of elements of Σ (Σ_0, Σ_1, respectively) and their negations such that for every $E \in \Sigma$ ($E \in \Sigma_0$, Σ_1, respectively) exactly one of E and $\neg E$ occurs in τ.
Note that every Σ_0-type (Σ_1-type) can be expressed as a union of Σ-types. Moreover, the number of Σ-types is is at most single exponential in the size of $\mathcal{E}$.

Consider a set expression $F(X_1, \ldots, X_n) \in \Sigma_1$. Let τ be a Σ_1-type, and let $\tau_1, \ldots, \tau_n$ be Σ_0-types such that $\tau \subseteq F(X_1, \ldots, X_n)$ and $\tau_i \subseteq X_i$, for $1 \leq i \leq n$.

We say that $(\tau_1, \dots, \tau_n)$ is a *domain* of τ (τ is an *image* of $(\tau_1, \dots, \tau_n)$) for $F(X_1, \dots, X_n)$ if

$$(\forall Y_1, \dots, Y_n \in \Sigma_0)(\tau_i \subseteq Y_i,\, 1 \leq i \leq n \quad \rightarrow \quad \tau \subseteq F(Y_1, \dots, Y_n)),$$

which we write as

$$(\tau_1, \dots, \tau_n) \overset{F(X_1,\dots,X_n)}{\rightarrow} \tau.$$

The expression $(\tau_1, \dots, \tau_n) \overset{F(X_1,\dots,X_n)}{\rightarrow} \tau$ denotes that it is possible that in a structure $\mathcal{M}$ there exist $x_1, \dots, x_n$ belonging to $\tau_1^{\mathcal{M}}, \dots, \tau_n^{\mathcal{M}}$, respectively such that $F^{\mathcal{M}}(x_1, \dots, x_n) \subseteq \tau^{\mathcal{M}}$. Denote by $\Delta_{F(X_1,\dots,X_n)}^{(\tau_1,\dots,\tau_n),\tau}$ the number such n-tuples $x_1, \dots, x_n$. Formally:

$$\Delta_{F(X_1,\dots,X_n)}^{(\tau_1,\dots,\tau_n),\tau} = \#\left\{(x_1, \dots, x_n) \,|\, x_i \in \tau_i^{\mathcal{M}},\, 1 \leq i \leq n,\, F^{\mathcal{M}}(x_1, \dots, x_n) \subseteq \tau^{\mathcal{M}}\right\}.$$

Given a relational structure $\mathcal{M}$, the description of $\mathcal{M}$ is the pair $\langle Card, \Delta \rangle$, where $Card$ is a function giving for a Σ-type τ the number $Card(\tau)$ of elements realizing τ in $\mathcal{M}$ (i.e the number of such x that $x \in \tau^{\mathcal{M}}$), and Δ is a function giving, for each $F(X_1, \dots, X_n)$ and $(\tau_1, \dots, \tau_n), \tau$ the cardinality of $\Delta_{F(X_1,\dots,X_n)}^{(\tau_1,\dots,\tau_n),\tau}$. For a given pair $\langle Card, \Delta \rangle$ as above, to determine whether it is a description of a structure it suffices to check the *non-emptiness conditions* and the *cardinality conditions* defined below.

Definition 4. *The* non-emptiness conditions *hold if*

(1) For every $F(X_1, \dots, X_n) \in \Sigma_1$ and a non-empty Σ_1-type $\tau \subseteq F(X_1, \dots, X_n)$ there exist non-empty Σ_0-types $\tau_i \subseteq X_i$, for $1 \leq i \leq n$ such that $(\tau_1, \dots, \tau_n) \overset{F(X_1,\dots,X_n)}{\rightarrow} \tau$.

(2) For every $f(X_1, \dots, X_n) \in \Sigma_1$, where f is a function and all non-empty Σ_0-types $\tau_i \subseteq X_i$, $1 \leq i \leq n$ there exists a non-empty Σ_1-type τ such that $(\tau_1, \dots, \tau_n) \overset{f(X_1,\dots,X_n)}{\rightarrow} \tau$.

The cardinality conditions *hold if*

(1) $\sum_{\tau \subseteq c} Card(\tau) = 1$, for every constant symbol c.

(2) For each function symbol f, each Σ_1-type τ, all variables X_i and Σ_0-types τ_i such that $\tau_i \subseteq X_i$, for $1 \leq i \leq n$,

$$\sum_{\tau} \Delta_{f(X_1,\dots,X_n)}^{(\tau_1,\dots,\tau_n),\tau} = \prod_{1 \leq i \leq n} Card(\tau_i).$$

(3) For each function symbol f, each Σ_1-type τ and all variables X_i

$$\sum_{(\tau_1,\dots,\tau_n)} \Delta_{f(X_1,\dots,X_n)}^{(\tau_1,\dots,\tau_n),\tau} \geq Card(\tau).$$

Part (1) of non-emptiness conditions says that if a function has a non-empty image, then it has a non-empty domain and part (2) says the converse. Part (2) of the cardinality conditions says that every member of domain has exactly one image and part (3) that every member of image is the unique image for at least one member of domain.

4 Main Theorem

Consider a system S of set constraints of size n satisfiable in a structure $\mathcal{M}$. In fact, as we have noticed earlier, we can assume that S is without variables, and we do it. So, we assume that $\mathcal{M}$ is a model of S. We shall construct another model $\mathcal{M}'$ of S in which the cardinality of each set defined by a Σ-type is either infinite or at most double exponential in n. A description of $\mathcal{M}'$ can be written in single exponential space.
We distinguish three kinds of Σ-types. The Σ-types in the first group we call *empty*, in the second one — *finite*, in the third one — *infinite*.
The set of *empty* Σ-types depends on structure we have chosen as original (in our case on $\mathcal{M}$). A Σ-type is *empty* if is contained in Σ_0-type or Σ_1-type which is empty in $\mathcal{M}$. Denote by $\Sigma_\emptyset^{\mathcal{M}}$ the set of all *empty* Σ-types for $\mathcal{M}$.

$$\Sigma_\emptyset^{\mathcal{M}} = \{\tau' \mid \exists_\tau \tau \subseteq \tau,\, \tau^{\mathcal{M}} = \emptyset\}$$

where τ is Σ_0-type or Σ_1-type.
In order to find *finite* Σ-types we use the obvious fact that if $X_1, \ldots, X_k$ define finite sets and f is a function then $f(X_1, \ldots, X_k)$ is finite. Note that for non-deterministic operators this is not true.
The Σ-types, which are neither *empty* nor *finite* are *infinite*.
Now we describe a procedure which determines Σ-types which are *finite*. This procedure works in steps. To provide some intuitions we give an example.
Let S be the system

$$X \subseteq a \cup c, \quad Y \subseteq f(X, X), \quad W \subseteq f(Y, Z),$$

where a, c are constants, X, Y, W, Z are set variables, and f is a function.
In the preprocessing step we mark as *finite* Σ-types which are always empty by given system. In our example we mark all Σ-types contained in $X \cap \neg(a \cup c)$, $Y \cap \neg f(X, X)$ or $W \cap \neg f(Y, Z)$.
During each step we mark as finite an image of a function for which the domain has been marked earlier as finite and we stop if we can not continue this procedure. In our example a and c will be marked in the first step since the domain of a constant is empty. In fact X (all types containing X) will be marked since $X \subseteq (X \cap \neg(a \cup c)) \cup (X \cap (a \cup c))$ - the first summand being finite by preprocessing and the second being included in $a \cup c$. In the second step we mark Y and the procedure stops after two steps. So, if S is satisfied, the cardinality of X is at most 2, and the cardinality of Y - at most 4. There is no reason, however, to bound the cardinalities of W and Z, so we can assume that they are infinite.

The proof of the main result of this part goes as follows. First we prove that for a system S of set constraints of size n the procedure outlined above stops after n steps and gives a set $\mathcal{F}_n$ of types. Then we use the fact that the cardinality of an image does not exceed the cardinality of the domain to get the doubly exponential bound on the size of sets definable by types in $\mathcal{F}_n$. Finally given a model of S we change it by leaving the types in $\mathcal{F}_n$ unchanged and making all other types infinite. The structure so obtained has a description of exponential size and is a model of S.

Definition 5. *Let $\mathcal{E}$ be a flat constraint.*
1. By $\mathcal{F}$ we denote the operator on the powerset of Σ-types defined as follows:

$$\mathcal{F}(\mathcal{P}) = \{\tau \in \Sigma \,|\, (\exists f(X_1,\dots,X_n) \in \Sigma_1)(\exists \tau_1 \in \Sigma_0)\dots(\exists \tau_n \in \Sigma_0) \\ (((\tau_1,\dots,\tau_n) \overset{f(X_1,\dots,X_n)}{\rightarrow} \tau) \wedge (\forall 1 \le i \le n)(\tau_i \subseteq \mathcal{P}))\},$$

where Σ_1-type τ represents all Σ-types contained in τ.
2. We put $\mathcal{F}_0 = \{\tau \,|\, \tau \subseteq \mathcal{E}\} \cup \Sigma_{\emptyset}^{\mathcal{M}}$, $\mathcal{F}_{n+1} = \mathcal{F}_n \cup \mathcal{F}(\mathcal{F}_n)$. By $\mathcal{F}_\infty$ we denote $\mathcal{F}_n$, for minimal n is such that $\mathcal{F}_{n+1} = \mathcal{F}_n$.

Part 1 of this Definition is quite technical but its meaning is quite clear. The operator $\mathcal{F}$ works as follows. In each step for any function f and a type τ which is the image under f of types which are all in $\mathcal{P}$, τ is included in $\mathcal{F}(\mathcal{P})$. In part 2 we include in $\mathcal{F}_0$ all types which are empty by direct application of given constraints or by original structure $\mathcal{M}$.

Lemma 1. *$\mathcal{F}_\infty = \mathcal{F}_n$, where n is the size of S.*

Proof. Clearly $\mathcal{F}_{n+1} = \mathcal{F}_n$ if each Σ_0 type belonging to $\mathcal{F}_n$ belongs already to $\mathcal{F}_{n-1}$. Moreover, if a Σ_0-type τ_1 is included into $\mathcal{F}$ before Σ_0-type τ_2 then there is a Σ_1-type τ such that $\tau_1 \cap \tau$ is in $\mathcal{F}_0$ and $\tau_2 \cap \tau$ is not in $\mathcal{F}_0$. To prove this assume that in a stage i the operator $\mathcal{F}$ adds a new Σ_0 type τ_1 and that τ_2 is not added neither in this stage nor has been added earlier. Since the only relationship between Σ_0 and Σ_1-types is through $\mathcal{F}_0$ it means that in the stage i a new Σ_1-type τ has been added and $\tau_1 \cap \neg\tau$ is in $\mathcal{F}_{i-1}$. Moreover, at least one of Σ-types contained $\tau_2 \cap \neg\tau$ is not in $\mathcal{F}_{i-1}$ since otherwise τ_2 would have been added in stage i.
Now, we will show that for a flat system $\mathcal{E} \subseteq \emptyset$ of set constraints of size n, if k is an integer for which there exist Σ_0-types $\tau_0,\dots,\tau_k$ and Σ_1-types $\tau'_1,\dots,\tau'_k$ such that

$$\tau_{i-1} \cap \tau'_i \in \mathcal{F}_0,\ \tau_i \cap \tau'_i \notin \mathcal{F}_0,\ \text{for } 1 \le i \le k,$$

then $k \le n$.
Note that the maximal value of k does not depend on $\Sigma_{\emptyset}^{\mathcal{M}}$. Recall that this set contains Σ-types corresponding to Σ_0-types and Σ_1-types. Then if $\tau_{i-1} \cap \tau'_i \in \Sigma_{\emptyset}^{\mathcal{M}}$, for some i then $\tau_{i-1} \cap \tau'_j \in \Sigma_{\emptyset}^{\mathcal{M}}$ for $1 \le j \le k$ or $\tau_j \cap \tau'_i \in \Sigma_{\emptyset}^{\mathcal{M}}$, for every $0 \le j \le k-1$. So the last condition can be replaced by

$$\tau_{i-1} \cap \tau'_i \subseteq \mathcal{E},\ \tau_i \cap \tau'_i \not\subseteq \mathcal{E},\ \text{for } 1 \le i \le k,$$

We prove it by induction on the size of $\mathcal{E}$.
We consider several cases:
Case 1. $\mathcal{E}$ has the form X.
Then for every Σ_0-type τ either τ is contained in $\mathcal{E}$ or τ and $\mathcal{E}$ are disjoint. It is clear that $k \leq 1$.
Case 2. $\mathcal{E}$ has the form $F(X_1, \ldots, X_n)$.
Then for every Σ_1-type τ' either τ' is contained in $\mathcal{E}$ or τ' and $\mathcal{E}$ are disjoint and therefore $k \leq 0$.
Case 3. $\mathcal{E}$ has the form $\mathcal{E} = \mathcal{E}_1 \cup \mathcal{E}_2$.
Then for each Σ_0-type τ and each Σ_1-type τ' if $\tau \cap \tau' \subseteq \mathcal{E}$ then $\tau \cap \tau' \subseteq \mathcal{E}_1$ or $\tau \cap \tau' \subseteq \mathcal{E}_2$. Moreover, for each Σ_0-type τ and each Σ_1-type τ' if $\tau \cap \tau' \not\subseteq \mathcal{E}$ then $\tau \cap \tau' \not\subseteq \mathcal{E}_1$ and $\tau \cap \tau' \not\subseteq \mathcal{E}_2$. From this it follows that $k \leq k_1 + k_2$, where k_1 and k_2 are the maximal values for $\mathcal{E}_1$ and $\mathcal{E}_2$, respectively.
Case 4. $\mathcal{E}$ has the form $\mathcal{E}_1 \cap \mathcal{E}_2$.
Then for each Σ_0-type τ and each Σ_1-type τ' if $\tau \cap \tau' \subseteq \mathcal{E}$ then $\tau \cap \tau' \subseteq \mathcal{E}_1$ and $\tau \cap \tau' \subseteq \mathcal{E}_2$. Similarly for $\tau \cap \tau' \not\subseteq \mathcal{E}$ we have $\tau \cap \tau' \not\subseteq \mathcal{E}_1$ or $\tau \cap \tau' \not\subseteq \mathcal{E}_2$. From this we deduce that $k \leq k_1 + k_2$, where k_1 and k_2 are the maximal values for $\mathcal{E}_1$ and $\mathcal{E}_2$, respectively.
This completes the proof of Lemma. $\square$

Lemma 2. *In every structure $\mathcal{M}$ each type in $\mathcal{F}_n$ is realized by at most doubly exponential number of elements of M.*

Proof. We use the fact that $Card(f(X_1, \ldots, X_n)) \leq \prod_{1 \leq i \leq n} Card(X_i)$, and the fact that the arity of all function symbols in $\mathcal{E}$ is bounded. $\square$

Theorem 1. *If a system S of set constraints is satisfiable by a structure $\mathcal{M}$ then S is satisfiable by a structure $\mathcal{M}'$ such that the cardinality of every Σ-type interpreted in $\mathcal{M}'$ is infinite or at most double exponential in the size of S.*

Proof. By Proposition 1 we can assume that S is of the form $\mathcal{E} \subseteq \emptyset$ where $\mathcal{E}$ is a flat expression.
To construct a structure $\mathcal{M}'$ from $\mathcal{M}$ we keep all types in $\mathcal{F}_\infty$ unchanged, and make all other types infinite. So be the Lemma 2 of all types in $\mathcal{F}_\infty$ have cardinality at most double exponential.
Moreover, put $\Delta^{(\tau_1, \ldots, \tau_n), \tau}_{F(X_1, \ldots, X_m)} = \infty$ if $\tau_i = \infty$, for some $1 \leq i \leq n$, and leave it unchanged otherwise. Clearly, as in the case of cardinality of types, the Δ is either infinite or at most double exponential. It is easy to check that the non-emptiness and cardinality conditions remain valid after these changes. $\square$

By the Theorem 1 to determine satisfiability of finite set of constraints it suffices to guess an exponential description D_S of a structure to and verify correctness of D_S i.e. the non-emptiness and the cardinality conditions for D_S.

References

1. A. Aiken, D. Kozen, M. Vardi, and E. L. Wimmers. The complexity of set constraints. In *Computer Science Logic'93*, LNCS 832, pages 1–17. Springer-Verlag, 1994.

2. A. Aiken, D. Kozen, and E. L. Wimmers. Decidability of systems of set constraints with negative constraints. Technical Report 93-1362, Computer Science Department, Cornell University, June 1993.
3. A. Aiken and B. Murphy. Static type inference in a dynamically typed language. In *Eighteenth Annual ACM Symposium on Principles of Programming Languages*, pages 279–290, January 1991.
4. A. Aiken and E. L. Wimmers. Solving systems of set constraints (extended abstract). In *Seventh Annual IEEE Symposium on Logic in Computer Science*, pages 329–340, 1992.
5. L. Bachmair, H. Ganzinger, and U. Waldmann. Set constraints are the monadic class. In *Eight Annual IEEE Symposium on Logic in Computer Science*, pages 75–83, 1993.
6. W. Charatonik and L. Pacholski. Set constraints with projections are in nexptime. In *35th Annual IEEE Symposium on Foundations of Computer Science*, pages 642–653, November 1994.
7. R. Gilleron, S. Tison, and M. Tommasi. Solving systems of set constraints using tree automata. In *10th Annual Symposium on Theoretical Aspects of Computer Science*, LNCS 665, pages 505–514. Springer-Verlag, 1993.
8. R. Gilleron, S. Tison, and M. Tommasi. Solving systems of set constraints with negated subset relationships. In *Proceedings of the* 34^{th} *Symp. on Foundations of Computer Science*, pages 372–380, 1993. A full version *Technical report IT 247, Laboratoire d'Informatique Fondamentale de Lille*.
9. D. Givan, R. McAllester. New results on local inference relations. In M. K. Press, editor, *Principles of Knowledge representation and Reasoning: Proceedings of the Third International Conference*, pages 403–412, 1992.
10. N. Heintze and J. Jaffar. A decision procedure for a class of set constraints (extended abstract). In *Fifth Annual IEEE Symposium on Logic in Computer Science*, pages 42–51, 1990.
11. N. Heintze and J. Jaffar. A finite presentation theorem for approximating logic programs. In *Seventeenth Annual ACM Symposium on Principles of Programming Languages*, pages 197–209, January 1990.
12. N. D. Jones and S. S. Muchnick. Flow analysis and optimization of lisp-like structures. In *Sixth Annual ACM Symposium on Principles of Programming Languages*, pages 244–256, January 1979.
13. D. A. McAllester and R. Givan. Taxonomic syntax for first order inference. *Journal of ACM*, 40:346–283, 1993.
14. D. A. McAllester, R. Givan, C. Witty, and D. Kozen. Tarskian set constraints. In *Proceedings, 11*$^{\text{th}}$ *Annual IEEE Symposium on Logic in Computer Science*, pages 138–147, New Brunswick, New Jersey, July 1996. IEEE Computer Society Press.
15. P. Mishra and U. Reddy. Declaration-free type checking. In *Twelfth Annual ACM Symposium on the Principles of Programming Languages*, pages 7–21, 1985.
16. J. C. Reynolds. Automatic computation of data set definitions. *Information Processing*, 68:456–461, 1969.
17. K. Stefansson. Systems of set constraints with negative constraints are *NEXPTIME*-complete. In *Ninth Annual IEEE Symposium on Logic in Computer Science*, pages 137–141, 1994.
18. J. Young and P. O'Keefe. Experience with a type evaluator. In D. Bjørner, A. P. Ershov, and N. D. Jones, editors, *Partial Evaluation and Mixed Computation*, pages 573–581. North-Holland, 1988.